

CONTENTS

1 Welcome to scikitlearn 1

11 Installing scikitlearn 1

12 Frequently Asked Questions 2

13 Support 8

14 Related Projects 9

15 About us 12

16 Who is using scikitlearn 17

17 Release History 26

18 Version 0213 27

19 Version 0212 28

110 Version 0211 29

111 Version 0210 29

112 Version 0204 41

113 Version 0203 42

114 Version 0202 43

115 Version 0201 44

116 Version 0200 48

117 Previous Releases 63

118 Roadmap 145

119 Scikitlearn governance and decisionmaking 148

2 scikitlearn Tutorials 151

21 An introduction to machine learning with scikitlearn 151

22 A tutorial on statisticallearning for scientific data processing 157

23 Working With Text Data 186

24 Choosing the right estimator 193

25 External Resources Videos and Talks 194

3 User Guide 197

31 Supervised learning 197

32 Unsupervised learning 336

33 Model selection and evaluation 434

34 Inspection 573

35 Dataset transformations 575

36 Dataset loading utilities 625

37 Computing with scikitlearn 650

4 Glossary of Common Terms and API (Application Programming Interface) Elements 663

41 General Concepts 663

42 Class APIs and Estimator Types 672

| | | | |
|--|------|------|------|
| 43 Target Types | 674 | | |
| 44 Methods | 676 | | |
| 45 Parameters | 678 | | |
| 46 Attributes | 681 | | |
| 47 Data and sample properties | | 682 | |
| 5 Examples | 683 | | |
| 51 Miscellaneous examples | | 683 | |
| 52 Examples based on real world datasets | | | 716 |
| 53 Biclustering | 775 | | |
| 54 Calibration | 787 | | |
| 55 Classification | 805 | | |
| 56 Clustering | 820 | | |
| 57 Pipelines and composite estimators | | | 908 |
| 58 Covariance estimation | | 942 | |
| 59 Cross decomposition | | 957 | |
| 510 Dataset examples | | 961 | |
| 511 Decomposition | 970 | | |
| 512 Ensemble methods | | 1016 | |
| 513 Tutorial exercises | | 1070 | |
| 514 Feature Selection | | 1079 | |
| 515 Gaussian Process for Machine Learning | | | 1090 |
| 516 Missing Value Imputation | | 1119 | |
| 517 Inspection | 1125 | | |
| 518 Generalized Linear Models | | 1130 | |
| 519 Manifold learning | | 1214 | |
| 520 Gaussian Mixture Models | | 1244 | |
| 521 Model Selection | 1261 | | |
| 522 Multioutput methods | | 1311 | |
| 523 Nearest Neighbors | | 1314 | |
| 524 Neural Networks | | 1347 | |
| 525 Preprocessing | 1360 | | |
| 526 Semi Supervised Classification | | 1386 | |
| 527 Support Vector Machines | | 1399 | |
| 528 Working with text documents | | 1431 | |
| 529 Decision Trees | 1447 | | |
| 6 API (Application Programming Interface) Reference | 1459 | | |
| 61sklearnbase Base classes and utility functions | | | 1459 |
| 62sklearncalibration Probability Calibration | | 1467 | |
| 63sklearncluster Clustering | 1471 | | |
| 64sklearnclusterbicluster Biclustering | | 1517 | |
| 65sklearncompose Composite Estimators | | 1524 | |
| 66sklearncovariance Covariance Estimators | | 1532 | |
| 67sklearncrossdecomposition Cross decomposition | | | 1563 |
| 68sklearndatasets Datasets | 1577 | | |
| 69sklearndecomposition Matrix Decomposition | | 1622 | |
| 610sklearndiscriminantanalysis Discriminant Analysis | | | 1678 |
| 611sklearnndummy Dummy estimators | | 1685 | |
| 612sklearnensemble Ensemble Methods | | 1690 | |
| 613sklearnexceptions Exceptions and warnings | | 1730 | |
| 614sklearnexperimental Experimental | | 1735 | |
| 615sklearnfeatureextraction Feature Extraction | | 1736 | |
| 616sklearnfeatureselection Feature Selection | | 1764 | |

| | | | | |
|--------------|--------------------------------------|---------------------|---|------|
| 617 | sklearn | gaussianprocess | Gaussian Processes | 1798 |
| 618 | sklearn | isotonic | Isotonic regression | 1838 |
| 619 | sklearn | impute | Impute | 1843 |
| 620 | sklearn | kernelapproximation | Kernel Approximation | 1851 |
| 621 | sklearn | kernelridge | Kernel Ridge Regression | 1861 |
| 622 | sklearn | linearmodel | Generalized Linear Models | 1864 |
| 623 | sklearn | manifold | Manifold Learning | 1965 |
| 624 | sklearn | metrics | Metrics | 1983 |
| 625 | sklearn | mixture | Gaussian Mixture Models | 2056 |
| 626 | sklearn | modelselection | Model Selection | 2068 |
| 627 | sklearn | multiclass | Multiclass and multilabel classification | 2122 |
| 628 | sklearn | multioutput | Multioutput regression and classification | 2130 |
| 629 | sklearn | naivebayes | Naive Bayes | 2140 |
| 630 | sklearn | neighbors | Nearest Neighbors | 2153 |
| 631 | sklearn | neuralnetwork | Neural network models | 2205 |
| 632 | sklearn | pipeline | Pipeline | 2218 |
| 633 | sklearn | inspection | inspection | 2227 |
| 634 | sklearn | preprocessing | Preprocessing and Normalization | 2231 |
| 635 | sklearn | randomprojection | Random projection | 2286 |
| 636 | sklearn | semisupervised | SemiSupervised Learning | 2293 |
| 637 | sklearn | svm | Support Vector Machines | 2299 |
| 638 | sklearn | tree | Decision Trees | 2331 |
| 639 | sklearn | utils | Utilities | 2358 |
| 640 | Recently deprecated | | | 2386 |
| 7 | Developer's Guide | | | 2407 |
| 71 | Contributing | | | 2407 |
| 72 | Developers' Tips and Tricks | | | 2429 |
| 73 | Utilities for Developers | | | 2433 |
| 74 | How to optimize for speed | | | 2436 |
| 75 | Advanced installation instructions | | | 2443 |
| 76 | Maintainer coredeveloper information | | | 2447 |
| Bibliography | | | | 2451 |
| Index | | | | 2459 |

iii

CHAPTER

ONE

WELCOME TO SCIKITLEARN

11 Installing scikitlearn

Note If you wish to contribute to the project it's recommended you install the latest development version

111 Installing the latest release

Scikitlearn requires

- Python 35
- NumPy 1110
- SciPy 0170
- joblib 011

Scikitlearn plotting capabilities ie functions start with " plot " require Matplotlib 151 Some of the scikit learn examples might require one or more extra dependencies scikitimage 0123 pandas 0180

Warning Scikitlearn 020 was the last version to support Python 27 and Python 34 Scikitlearn now requires Python 35 or newer

If you already have a working installation of numpy and scipy the easiest way to install scikitlearn is using pip
pip install U scikitlearn

orconda
conda install scikitlearn

If you have not installed NumPy or SciPy yet you can also install these using conda or pip When using pip please ensure that binary wheels are used and NumPy and SciPy are not recompiled from source which can happen when using particular configurations of operating system and hardware such as Linux on a Raspberry Pi Building numpy and scipy from source can be complex especially on Windows and requires careful configuration to ensure that they link against an optimized implementation of linear algebra routines Instead use a thirdparty distribution as described below

scikitlearn user guide Release 0.21.3

If you must install scikitlearn and its dependencies with pip you can install it as scikitlearnalldeps. The most common use case for this is in a requirements.txt file used as part of an automated build process for a PaaS application or a Docker image. This option is not intended for manual installation from the command line. Note: For installing on PyPy, PyPy3v5.10, NumPy 1.14.0 and scipy 1.1.0 are required. For installation instructions for more distributions see other distributions. For compiling the development version from source or building the package if no distribution is available for your architecture see the Advanced installation instructions.

1.1.2 Thirdparty Distributions

If you don't already have a python installation with numpy and scipy we recommend to install either via your package manager or via a python bundle. These come with numpy, scipy, scikitlearn, matplotlib and many other helpful scientific and data processing libraries.

Available options are:

Canopy and Anaconda for all supported platforms.

Canopy and Anaconda both ship a recent version of scikitlearn in addition to a large set of scientific python library for Windows, Mac, OSX and Linux.

Anaconda offers scikitlearn as part of its free distribution.

Warning: To upgrade or uninstall scikitlearn installed with Anaconda or conda you should not use the pip command. Instead:

To upgrade scikitlearn:

conda update scikitlearn

To uninstall scikitlearn:

conda remove scikitlearn

Upgrading with pip: install U scikitlearn or uninstalling pip: uninstall scikitlearn is likely to fail to properly remove files installed by the conda command.

pip upgrade and uninstall operations only work on packages installed via pip install.

WinPython for Windows

The WinPython project distributes scikitlearn as an additional plugin.

1.2 Frequently Asked Questions

Here we try to give some answers to questions that regularly pop up on the mailing list.

2 Chapter 1: Welcome to scikitlearn

scikitlearn user guide Release 0.21.3

121 What is the project name a lot of people get it wrong
scikitlearn but not scikit or SciKit nor scikit learn Also not scikitslearn or scikitslearn which were previously used

122 How do you pronounce the project name
syk-it learn sci stands for science

123 Why scikit
There are multiple scikits which are scientific toolboxes built around SciPy You can find a list at <http://ssci.kits>
appspot.com/scikits Apart from scikitlearn another popular one is scikitimage

124 How can I contribute to scikitlearn
See Contributing Before wanting to add a new algorithm which is usually a major and lengthy undertaking it is recommended to start with known issues Please do not contact the contributors of scikitlearn directly regarding contributing to scikitlearn

125 What's the best way to get help on scikitlearn usage
For general machine learning questions please use Cross Validated with the machinelearning tag
For scikitlearn usage questions please use Stack Overflow with the scikitlearn andpython tags You can alternatively use the mailing list

Please make sure to include a minimal reproduction code snippet ideally shorter than 10 lines that highlights your problem on a toy dataset for instance from sklearn.datasets or randomly generated with functions of numpy random with a fixed random seed Please remove any line of code that is not necessary to reproduce your problem The problem should be reproducible by simply copy-pasting your code snippet in a Python shell with scikitlearn installed Do not forget to include the import statements

More guidance to write good reproduction code snippets can be found at <https://stackoverflow.com/help/mcve>

If your problem raises an exception that you do not understand even after googling it please make sure to include the full traceback that you obtain when running the reproduction script

For bug reports or feature requests please make use of the issue tracker on GitHub

There is also a scikitlearn Gitter channel where some users and developers might be found

Please do not email any authors directly to ask for assistance report bugs or for any other issue related to scikitlearn

126 How should I save export or deploy estimators for production

See Model persistence

12 Frequently Asked Questions 3

scikitlearn user guide Release 0213

127 How can I create a bunch object

Don't make a bunch object They are not part of the scikitlearn API (Application Programming Interface) Bunch objects are just a way to package so
numpy arrays As a scikitlearn user you only ever need numpy arrays to feed your model with data

For instance to train a classifier all you need is a 2D array Xfor the input variables and a 1D array yfor the target
variables The array Xholds the features as columns and samples as rows The array ycontains integer values to
encode the class membership of each sample in X

128 How can I load my own datasets into a format usable by scikitlearn

Generally scikitlearn works on any numeric data stored as numpy arrays or scipy sparse matrices Other types that
are convertible to numeric arrays such as pandas DataFrame are also acceptable

For more information on loading your data files into these usable data structures please refer to loading external
datasets

129 What are the inclusion criteria for new algorithms

We only consider well-established algorithms for inclusion A rule of thumb is at least 3 years since publication 200
citations and wide use and usefulness A technique that provides a clearcut improvement eg an enhanced data
structure or a more efficient approximation technique on a widely-used method will also be considered for inclusion

From the algorithms or techniques that meet the above criteria only those which fit well within the current API (Application Programming Interface)

scikitlearn that is a fitpredicttransform interface and ordinarily having inputoutput that is a numpy array
or sparse matrix are accepted

The contributor should support the importance of the proposed addition with research papers and/or implementations
in other similar packages demonstrate its usefulness via common use-cases/applications and corroborate performance
improvements if any with benchmarks and/or plots It is expected that the proposed algorithm should outperform the
methods that are already implemented in scikitlearn at least in some areas

Inclusion of a new algorithm speeding up an existing model is easier if

- it does not introduce new hyperparameters as it makes the library more future-proof
- it is easy to document clearly when the contribution improves the speed and when it does not for instance “when
n-features n-samples”
- benchmarks clearly show a speed up

Also note that your implementation need not be in scikitlearn to be used together with scikitlearn tools You can
implement your favorite algorithm in a scikitlearn-compatible way upload it to GitHub and let us know We will be
happy to list it under Related Projects If you already have a package on GitHub following the scikitlearn API (Application Programming Interface) y
may also be interested to look at scikitlearn-contrib

1210 Why are you so selective on what algorithms you include in scikitlearn

Code is maintenance cost and we need to balance the amount of code we have with the size of the team and add to
this the fact that complexity scales non-linearly with the number of features The package relies on core developers
using their free time to fix bugs maintain code and review contributions Any algorithm that is added needs future
attention by the developers at which point the original author might long have lost interest See also What are the
inclusion criteria for new algorithms For a great read about long-term maintenance issues in open-source software
look at the Executive Summary of Roads and Bridges

4 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

1211 Why did you remove HMMs from scikitlearn

SeeWill you add graphical models or sequence prediction to scikitlearn

1212 Will you add graphical models or sequence prediction to scikitlearn

Not in the foreseeable future scikitlearn tries to provide a unified API (Application Programming Interface) for the basic tasks in machine learning w
pipelines and metaalgorithms like grid search to tie everything together The required concepts APIs algorithms
and expertise required for structured learning are different from what scikitlearn has to offer If we started doing
arbitrary structured learning we'd need to redesign the whole package and the project would likely collapse under its
own weight

There are two project with API (Application Programming Interface) similar to scikitlearn that do structured prediction

- pystruct handles general structured learning focuses on SSVMs on arbitrary graph structures with approximate inference defines the notion of sample as an instance of the graph structure
- seqlearn handles sequences only focuses on exact inference has HMMs but mostly for the sake of complete ness treats a feature vector as a sample and uses an offset encoding for the dependencies between feature vectors

1213 Will you add GPU support

No or at least not in the near future The main reason is that GPU support will introduce many software dependencies and introduce platform specific issues scikitlearn is designed to be easy to install on a wide variety of platforms Outside of neural networks GPUs don't play a large role in machine learning today and much larger gains in speed can often be achieved by a careful choice of algorithms

1214 Do you support PyPy

In case you didn't know PyPy is an alternative Python implementation with a builtin justintime compiler Experi mental support for PyPy3v510 has been added which requires Numpy 1140 and scipy 110

1215 How do I deal with string data or trees graphs

scikitlearn estimators assume you'll feed them realvalued feature vectors This assumption is hardcoded in pretty much all of the library However you can feed nonnumerical inputs to estimators in several ways

If you have text documents you can use a term frequency features see Text feature extraction for the builtin text vectorizers For more general feature extraction from any kind of data see Loading features from dicts andFeature hashing

Another common case is when you have nonnumerical data and a custom distance or similarity metric on these data Examples include strings with edit distance aka Levenshtein distance eg DNA or RNA sequences These can be encoded as numbers but doing so is painful and errorprone Working with distance metrics on arbitrary data can be done in two ways

Firstly many estimators take precomputed distancesimilarity matrices so if the dataset is not too large you can compute distances for all pairs of inputs If the dataset is large you can use feature vectors with only one "feature" which is an index into a separate data structure and supply a custom metric function that looks up the actual data in this data structure Eg to use DBSCAN with Levenshtein distances

12 Frequently Asked Questions 5

```
scikitlearn user guide Release 0213
from leven import levenshtein
import numpy as np
from sklearncluster import dbscan
data ACCTCCTAGAAG ACCTACTAGAAGTT GAATATTAGGCCGA
def levmetricx y
i j intx0 inty0 extract indices
return levenshteindatai dataj
```

```
X nparangelendatareshape1 1
X
array0
1
2
```

We need to specify algoritumbrute as the default assumes
a continuous feature space
dbscanX metriclevmetric eps5 minsamples2 algorithmbrute

```
0 1 array 0 0 1
```

This uses the thirdparty edit distance package leven
Similar tricks can be used with some care for tree kernels graph kernels etc
1216 Why do I sometime get a crashfreeze with njobs 1 under OSX or Linux
Several scikitlearn tools such as GridSearchCV andcrossvalscore rely internally on Python’s
multiprocessing module to parallelize execution onto several Python processes by passing njobs 1 as
argument

The problem is that Python multiprocessing does afork system call without following it with an exec system
call for performance reasons Many libraries like some versions of Accelerate vecLib under OSX some versions
of MKL the OpenMP runtime of GCC nvidia’s Cuda and probably many others manage their own internal thread
pool Upon a call to fork the thread pool state in the child process is corrupted the thread pool believes it has many
threads while only the main thread state has been forked It is possible to change the libraries to make them detect
when a fork happens and reinitialize the thread pool in that case we did that for OpenBLAS merged upstream in
master since 0210 and we contributed a patch to GCC’s OpenMP runtime not yet reviewed

But in the end the real culprit is Python’s multiprocessing that doesfork withoutexec to reduce the overhead
of starting and using new Python processes for parallel computing Unfortunately this is a violation of the POSIX
standard and therefore some software editors like Apple refuse to consider the lack of forksafety in Accelerate
vecLib as a bug

In Python 34 it is now possible to configure multiprocessing to use the ‘forkserver’ or ‘spawn’ start methods
instead of the default ‘fork’ to manage the process pools To work around this issue when using scikitlearn you
can set the JOBLIBSTARTMETHOD environment variable to ‘forkserver’ However the user should be aware that
using the ‘forkserver’ method prevents joblibParallel to call function interactively defined in a shell session
If you have custom code that uses multiprocessing directly instead of using it via joblib you can enable the
‘forkserver’ mode globally for your program Insert the following instructions in your main script

```
import multiprocessing
other imports custom code load data define model
ifname main
multiprocessingsetstartmethodforkserver
```

scikitlearn user guide Release 0213

call scikitlearn utils with njobs 1 here

You can find more default on the new start methods in the multiprocessing documentation

1217 Why does my job use more cores than specified with njobs under OSX or

Linux

This happens when vectorized numpy operations are handled by libraries such as MKL or OpenBLAS

While scikitlearn adheres to the limit set by njobs numpy operations vectorized using MKL or OpenBLAS will

make use of multiple threads within each scikitlearn job thread or process

The number of threads used by the BLAS library can be set via an environment variable For example to set the

maximum number of threads to some integer value N the following environment variables should be set

- For MKL export MKLNUMTHREADSN
- For OpenBLAS export OPENBLASNUMTHREADSN

1218 Why is there no support for deep or reinforcement learning Will there be

support for deep or reinforcement learning in scikitlearn

Deep learning and reinforcement learning both require a rich vocabulary to define an architecture with deep learning

additionally requiring GPUs for efficient computing However neither of these fit within the design constraints of

scikitlearn as a result deep learning and reinforcement learning are currently out of scope for what scikitlearn seeks

to achieve

You can find more information about addition of gpu support at Will you add GPU support

1219 Why is my pull request not getting any attention

The scikitlearn review process takes a significant amount of time and contributors should not be discouraged by a

lack of activity or review on their pull request We care a lot about getting things right the first time as maintenance

and later change comes at a high cost We rarely release any “experimental” code so all of our contributions will be

subject to high use immediately and should be of the highest quality possible initially

Beyond that scikitlearn is limited in its reviewing bandwidth many of the reviewers and core developers are working

on scikitlearn on their own time If a review of your pull request comes slowly it is likely because the reviewers are

busy We ask for your understanding and request that you not close your pull request or discontinue your work solely

because of this reason

1220 How do I set a randomstate for an entire execution

For testing and replicability it is often important to have the entire execution controlled by a single seed for the pseudo

random number generator used in algorithms that have a randomized component Scikitlearn does not use its own

global random state whenever a RandomState instance or an integer random seed is not provided as an argument it

relies on the numpy global random state which can be set using numpyrandomseed For example to set an

execution’s numpy global random state to 42 one could execute the following in his or her script

```
import numpy as np
```

```
np.random.seed(42)
```

12 Frequently Asked Questions 7

scikitlearn user guide Release 0213

However a global random state is prone to modification by other code during execution Thus the only way to ensure replicability is to pass RandomState instances everywhere and ensure that both estimators and crossvalidation splitters have their randomstate parameter set

1221 Why do categorical variables need preprocessing in scikitlearn compared to other tools

Most of scikitlearn assumes data is in NumPy arrays or SciPy sparse matrices of a single numeric dtype These do not explicitly represent categorical variables at present Thus unlike R's dataframes or pandasDataFrame we require explicit conversion of categorical features to numeric values as discussed in Encoding categorical features See also Column Transformer with Mixed Types for an example of working with heterogeneous eg categorical and numeric data

1222 Why does Scikitlearn not directly work with for example pandasDataFrame

The homogeneous NumPy and SciPy data objects currently expected are most efficient to process for most operations Extensive work would also be needed to support Pandas categorical types Restricting input to homogeneous types therefore reduces maintenance cost and encourages usage of efficient data structures

13 Support

There are several ways to get in touch with the developers

131 Mailing List

- The main mailing list is scikitlearn
- There is also a commit list scikitlearncommits where updates to the main repository and test failures get notified

132 User questions

- Some scikitlearn developers support users on StackOverflow using the scikitlearn tag
- For general theoretical or methodological Machine Learning questions stack exchange is probably a more suitable venue

In both cases please use a descriptive question in the title field eg no "Please help with scikitlearn" as this is not a question and put details on what you tried to achieve what were the expected results and what you observed instead in the details field

Code and data snippets are welcome Minimalistic up to 20 lines long reproduction script very helpful Please describe the nature of your data and the how you preprocessed it what is the number of samples what is the number and type of features id categorical or numerical and for supervised learning tasks what target are you trying to predict binary multiclass 1 out of nclasses or multilabel k out of nclasses classification or continuous variable regression

8 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

133 Bug tracker

If you think you’ve encountered a bug please report it to the issue tracker

<https://github.com/scikitlearn/scikitlearn/issues>

Don’t forget to include

- steps or better script to reproduce
- expected outcome
- observed outcome or python or gdb tracebacks

To help developers fix your bug faster please link to a <https://gist.github.com> holding a standalone minimalistic python script that reproduces your bug and optionally a minimalistic subsample of your dataset for instance exported as CSV files using `numpy.savetxt`

Note gists are git cloneable repositories and thus you can use git to push datafiles to them

134 IRC

Some developers like to hang out on channel `#scikitlearn` on `irc.freenode.net`

If you do not have an IRC client or are behind a firewall this web client works fine <https://webchat.freenode.net>

135 Documentation resources

This documentation is relative to 0213 Documentation for other versions can be found [here](#)

Printable pdf documentation for old versions can be found [here](#)

14 Related Projects

Projects implementing the scikitlearn estimator API (Application Programming Interface) are encouraged to use the `scikitlearncontrib` template which facilitates best practices for testing and documenting estimators The `scikitlearncontrib` GitHub organisation also

accepts highquality contributions of repositories conforming to this template

Below is a list of sisterprojects extensions and domain specific packages

141 Interoperability and framework enhancements

These tools adapt scikitlearn for use with other technologies or otherwise enhance the functionality of scikitlearn’s estimators

Data formats

- `sklearnpandas` bridge for scikitlearn pipelines and pandas data frame with dedicated transformers
- `sklearnxarray` provides compatibility of scikitlearn estimators with xarray data structures

AutoML

- `automl` Automated machine learning for production and analytics built on scikitlearn and related projects

Trains a pipeline with all the standard machine learning steps Tuned for prediction speed and ease of transfer to production environments

- `autosklearn` An automated machine learning toolkit and a dropin replacement for a scikitlearn estimator

14 Related Projects 9

scikitlearn user guide Release 0213

- TPOT An automated machine learning toolkit that optimizes a series of scikitlearn operators to design a machine learning pipeline including data and feature preprocessors as well as the estimators Works as a dropin replacement for a scikitlearn estimator
- scikitoptimize A library to minimize very expensive and noisy blackbox functions It implements several methods for sequential modelbased optimization and includes a replacement for GridSearchCV or RandomizedSearchCV to do crossvalidated parameter search using any of these strategies

Experimentation frameworks

- REP Environment for conducting datadriven research in a consistent and reproducible way
- ML Frontend provides dataset management and SVM (Support Vector Machine) fittingprediction through webbased and programmatic interfaces
- ScikitLearn Laboratory A commandline wrapper around scikitlearn that makes it easy to run machine learning experiments with multiple learners and large feature sets
- Xcessiv is a notebooklike application for quick scalable and automated hyperparameter tuning and stacked ensembling Provides a framework for keeping track of modelhyperparameter combinations

Model inspection and visualisation

- eli5 A library for debugginginspecting machine learning models and explaining their predictions
- mlxtend Includes model visualization utilities
- scikitplot A visualization library for quick and easy generation of common plots in data analysis and machine learning
- yellowbrick A suite of custom matplotlib visualizers for scikitlearn estimators to support visual feature analysis model selection evaluation and diagnostics

Model export for production

- onnxmltools Serializes many Scikitlearn pipelines to ONNX for interchange and prediction
- sklearn2pmmml Serialization of a wide variety of scikitlearn estimators and transformers into PMML with the help of JPMMLSkLearn library
- sklearnporter Transpile trained scikitlearn models to C Java Javascript and others
- sklearncompiledtrees Generate a C implementation of the predict function for decision trees and ensembles trained by sklearn Useful for latencysensitive production environments

142 Other estimators and tasks

Not everything belongs or is mature enough for the central scikitlearn project The following are projects providing interfaces similar to scikitlearn for additional learning algorithms infrastructures and tasks

Structured learning

- Seqlearn Sequence classification using HMMs or structured perceptron
- HMMLearn Implementation of hidden markov models that was previously part of scikitlearn
- PyStruct General conditional random fields and structured prediction
- pomegranate Probabilistic modelling for Python with an emphasis on hidden Markov models
- sklearncrfsuite Linearchain conditional random fields CRFsuite wrapper with sklearnlike API (Application Programming Interface)

Deep neural networks etc

- pylearn2 A deep learning and neural network library build on theano with scikitlearn like interface

10 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- sklearntheano scikitlearn compatible estimators transformers and datasets which use Theano internally
- nolearn A number of wrappers and abstractions around existing neural network libraries
- keras Deep Learning library capable of running on top of either TensorFlow or Theano
- lasagne A lightweight library to build and train neural networks in Theano
- skorch A scikitlearn compatible neural network library that wraps PyTorch

Broad scope

- mlxtend Includes a number of additional estimators as well as model visualization utilities
- sparkilearn Scikitlearn API (Application Programming Interface) and functionality for PySpark’s distributed modelling

Other regression and classification

- xgboost Optimised gradient boosted decision tree library
- MLEnsemble Generalized ensemble learning stacking blending subsemble deep ensembles etc
- lightning Fast stateoftheart linear model solvers SDCA AdaGrad SVRG SAG etc
- pyearth Multivariate adaptive regression splines
- Kernel Regression Implementation of NadarayaWatson kernel regression with automatic bandwidth selection
- gplearn Genetic Programming for symbolic regression tasks
- multiisotonic Isotonic regression on multidimensional features
- scikitmultilearn Multilabel classification with focus on label space manipulation
- seglearn Time series and sequence learning using sliding window segmentation

Decomposition and clustering

- lda Fast implementation of latent Dirichlet allocation in Cython which uses Gibbs sampling to sample from the true posterior distribution scikitlearn’s sklearndecomposition

LatentDirichletAllocation implementation uses variational inference to sample from a tractable approximation of a topic model’s posterior distribution

- Sparse Filtering Unsupervised feature learning based on sparsefiltering
- kmodes kmodes clustering algorithm for categorical data and several of its variations
- hdbscan HDBSCAN and Robust Single Linkage clustering algorithms for robust variable density clustering
- spherecluster Spherical Kmeans and mixture of von Mises Fisher clustering routines for data on the unit hyper sphere

Preprocessing

- categoricalencoding A library of sklearn compatible categorical variable encoders
- imbalancedlearn Various methods to under and oversample datasets

143 Statistical learning with Python

Other packages useful for data analysis and machine learning

- Pandas Tools for working with heterogeneous and columnar data relational queries time series and basic statistics
- theano A CPU/GPU array processing framework geared towards deep learning research

14 Related Projects 11

scikitlearn user guide Release 0213

- statsmodels Estimating and analysing statistical models More focused on statistical tests and less on prediction than scikitlearn

- PyMC Bayesian statistical models and fitting algorithms
- Sacred Tool to help you configure organize log and reproduce experiments
- Seaborn Visualization library based on matplotlib It provides a highlevel interface for drawing attractive statistical graphics

- Deep Learning A curated list of deep learning software libraries

Domain specific packages

- scikitimage Image processing and computer vision in python
- Natural language toolkit nltk Natural language processing and some machine learning
- gensim A library for topic modelling document indexing and similarity retrieval
- NiLearn Machine learning for neuroimaging
- AstroML Machine learning for astronomy
- MSMBUILDER Machine learning for protein conformational dynamics time series
- scikitsurprise A scikit for building and evaluating recommender systems

144 Snippets and tidbits

The wiki has more

15 About us

151 History

This project was started in 2007 as a Google Summer of Code project by David Cournapeau Later that year Matthieu

Brucher started work on this project as part of his thesis

In 2010 Fabian Pedregosa Gael Varoquaux Alexandre Gramfort and Vincent Michel of INRIA took leadership of the project and made the first public release February the 1st 2010 Since then several releases have appeared following a 3 month cycle and a thriving international community has been leading the development

152 Governance

The decision making process and governance structure of scikitlearn is laid out in the governance document

153 Authors

The following people are currently core contributors to scikitlearn’s development and maintenance

Please do not email the authors directly to ask for assistance or report issues Instead please see What’s the best way to ask questions about scikitlearn in the FAQ

See also

12 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213
How you can contribute to the project
154 Emeritus Core Developers

The following people have been active contributors in the past but are no longer active in the project

- Alexander Fabisch
- Alexandre Passos
- Angel Soler Gollonet
- Arnaud Joly
- Chris Gorgolewski
- David Cournapeau
- David WardeFarley
- Eduard Duchesnay
- Fabian Pedragosa
- Gilles Louppe
- Jacob Schreiber
- Jake Vanderplas
- Jaques Grobler
- Jarrod Millman
- Kyle Kastner
- Lars Buitinck
- Manoj Kumar
- Mathieu Blondel
- Matthieu Brucher
- Noel Dawe
- Paolo Losi
- Peter Prettenhofer
- Raghav Rajagopalan
- Robert Layton
- Ron Weiss
- Satrajit Ghosh
- Shiqiao Du
- Thouis Ray Jones
- Vincent Dubourg
- Vincent Michel
- Virgile Fritsch
- Wei Li

15 About us 13

scikitlearn user guide Release 0213

155 Citing scikitlearn

If you use scikitlearn in a scientific publication we would appreciate citations to the following paper

Scikitlearn Machine Learning in Python Pedregosa et al JMLR 12 pp 28252830 2011

Bibtex entry

article scikitlearn

titleScikitlearn Machine Learning inPython

authorPedregosa F andVaroquaux G andGramfort A andMichel V

andThirion B andGrisel O andBlondel M andPrettenhofer P

andWeiss R andDubourg V andVanderplas J andPassos A and

Cournapeau D andBrucher M andPerrot M andDuchesnay E

journalJournal of Machine Learning Research

volume12

pages28252830

year2011

If you want to cite scikitlearn for its API (Application Programming Interface) or design you may also want to consider the following paper

API (Application Programming Interface) design for machine learning software experiences from the scikitlearn project Buitinck et al 2013

Bibtex entry

inproceedings sklearnapi

author Lars Buitinck andGilles Louppe andMathieu Blondel and

Fabian Pedregosa andAndreas Mueller andOlivier Grisel and

Vlad Niculae andPeter Prettenhofer andAlexandre Gramfort

andJaques Grobler andRobert Layton andJake VanderPlas and

Arnaud Joly andBrian Holt andGael Varoquaux

title API (Application Programming Interface) design formachine learning software experiences from

↪thescikitlearn

project

booktitle ECML PKDD Workshop Languages forData Mining andMachine

↪Learning

year 2013

pages 108122

156 Artwork

High quality PNG and SVG logos are available in the doclogos source directory

14 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0.21.3

157 Funding

INRIA actively supports this project. It has provided funding for Fabian Pedregosa (2010-2012), Jaques Grobler (2012-2013) and Olivier Grisel (2013-2017) to work on this project fulltime. It also hosts coding sprints and other events.

Paris-Saclay Center for Data Science funded one year

for a developer to work on the project fulltime (2014-2015) and 50% of the time of Guillaume Lemaître (2016-2017).

NYU Moore-Sloan Data Science Environment funded Andreas

Mueller (2014-2016) to work on this project. The Moore-Sloan Data Science Environment also funds several students to work on the project parttime.

Télécom Paris

tech funded Manoj Kumar (2014), Tom Dupré la Tour (2015), Raghav RV (2015-2017), Thierry Guillemot (2016-2017) and Albert Thomas (2017) to work on scikitlearn.

Columbia University funds Andreas

Müller since 2016.

Andreas Müller also received a grant to improve scikitlearn from the Alfred P. Sloan Foundation in 2017.

The University of

15 About us 15

scikitlearn user guide Release 0213

Sydney funds Joel Nothman since July 2017

The Labex Digi

Cosme funded Nicolas Goix 20152016 Tom Dupré la Tour 20152016 and 20172018 Mathurin Massias 2018

2019 to work part time on scikitlearn during their PhDs It also funded a scikitlearn coding sprint in 2015

The following students were sponsored by Google to work on

scikitlearn through the Google Summer of Code program

- 2007 David Cournapeau
- 2011 Vlad Niculae
- 2012 Vlad Niculae Immanuel Bayer
- 2013 Kemal Eren Nicolas Trésegne
- 2014 Hamzeh Alsalmi Issam Laradji Maheshakya Wijewardena Manoj Kumar
- 2015 Raghav RV Wei Xue
- 2016 Nelson Liu YenChen Lin

It also provided funding for sprints and events around scikitlearn If you would like to participate in the next Google Summer of code program please see this page

The NeuroDebian project providing Debian packaging and contributions is supported by Dr James V Haxby Dartmouth College

The PSF helped find and manage funding for our 2011 Granada sprint More information can be found here

tinyclues funded the 2011 international Granada sprint

Donating to the project

If you are interested in donating to the project or to one of our codesprints you can use the Paypal button below or the

NumFOCUS Donations Page if you use the latter please indicate that you are donating for the scikitlearn project

All donations will be handled by NumFOCUS a nonprofitorganization which is managed by a board of Scipy

community members NumFOCUS's mission is to foster scientific computing software in particular in Python As

a fiscal home of scikitlearn it ensures that money is available when needed to keep the project funded and available

while in compliance with tax regulations

The received donations for the scikitlearn project mostly will go towards covering travelexpenses for code sprints as

well as towards the organization budget of the project1

1Regarding the organization budget in particular we might use some of the donated funds to pay for other project expenses such as DNS

hosting or continuous integration services

16 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

Notes

The 2013 Paris international sprint

Fig 11 IAP VII19 DYSCO

For more information on this sprint see here

158 Infrastructure support

- We would like to thank Rackspace for providing us with a free Rackspace Cloud account to automatically build the documentation and the example gallery from for the development version of scikitlearn using this tool
- We would also like to thank Shining Panda for free CPU time on their Continuous Integration server

16 Who is using scikitlearn

161 JPMorgan

Scikitlearn is an indispensable part of the Python machine learning toolkit at JPMorgan It is very widely used across all parts of the bank for classification predictive analytics and very many other machine learning tasks Its straightforward API (Application Programming Interface) its breadth of algorithms and the quality of its documentation combine to make scikitlearn simultaneously very approachable and very powerful

Stephen Simmons VP Athena Research JPMorgan

16 Who is using scikitlearn 17

scikitlearn user guide Release 0213

162 Spotify

Scikitlearn provides a toolbox with solid implementations of a bunch of stateoftheart models and makes it easy to plug them into existing applications We’ve been using it quite a lot for music recommendations at Spotify and I think it’s the most welldesigned ML package I’ve seen so far

Erik Bernhardsson Engineering Manager Music Discovery Machine Learning Spotify

163 Inria

At INRIA we use scikitlearn to support leadingedge basic research in many teams Parietal for neuroimaging Lear for computer vision Visages for medical image analysis Privatics for security The project is a fantastic tool to address difficult applications of machine learning in an academic environment as it is performant and versatile but all easytouse and well documented which makes it well suited to grad students

Gaël Varoquaux research at Parietal

164 betaworks

Betaworks is a NYCbased startup studio that builds new products grows companies and invests in others Over the past 8 years we’ve launched a handful of social data analyticsdriven services such as Bitly Chartbeat digg and Scale Model Consistently the betaworks data science team uses Scikitlearn for a variety of tasks From exploratory analysis to product development it is an essential part of our toolkit Recent uses are included in digg’s new video recommender system and Poncho’s dynamic heuristic subspace clustering

Gilad Lotan Chief Data Scientist

18 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

165 Hugging Face

At Hugging Face we're using NLP and probabilistic models to generate conversational Artificial intelligences that are fun to chat with Despite using deep neural nets for a few of our NLP tasks scikitlearn is still the breadandbutter of our daily machine learning routine The ease of use and predictability of the interface as well as the straightforward mathematical explanations that are here when you need them is the killer feature We use a variety of scikitlearn models in production and they are also operationally very pleasant to work with

Julien Chaumond Chief Technology Officer

166 Evernote

Building a classifier is typically an iterative process of exploring the data selecting the features the attributes of the data believed to be predictive in some way training the models and finally evaluating them For many of these tasks we relied on the excellent scikitlearn package for Python

Read more

Mark Ayzenshtat VP Augmented Intelligence

167 Télécom ParisTech

At Telecom ParisTech scikitlearn is used for handson sessions and home assignments in introductory and advanced machine learning courses The classes are for undergrads and masters students The great benefit of scikitlearn is its fast learning curve that allows students to quickly start working on interesting and motivating problems

16 Who is using scikitlearn 19

scikitlearn user guide Release 0213
Alexandre Gramfort Assistant Professor
168 Bookingcom

At Bookingcom we use machine learning algorithms for many different applications such as recommending hotels and destinations to our customers detecting fraudulent reservations or scheduling our customer service agents Scikitlearn is one of the tools we use when implementing standard algorithms for prediction tasks Its API (Application Programming Interface) and its implementations are excellent and make it easy to use The scikitlearn developers do a great job of incorporating state of the art implementations and new algorithms into the package Thus scikitlearn provides convenient access to a wide spectrum of algorithms and allows us to readily find the right tool for the right job

Melanie Mueller Data Scientist
169 AWeber

The scikitlearn toolkit is indispensable for the Data Analysis and Management team at AWeber It allows us to do awesome stuff we would not otherwise have the time or resources to accomplish The documentation is excellent allowing new engineers to quickly evaluate and apply many different algorithms to our data The text feature extraction utilities are useful when working with the large volume of email content we have at AWeber The RandomizedPCA implementation along with Pipelining and FeatureUnions allows us to develop complex machine learning algorithms efficiently and reliably

Anyone interested in learning more about how AWeber deploys scikitlearn in a production environment should check out talks from PyData Boston by AWeber's Michael Becker available at <https://github.com/mdbecker/pydata2013>
Michael Becker Software Engineer Data Analysis and Management Ninjas

1610 Yhat
The combination of consistent APIs thorough documentation and top notch implementation make scikitlearn our favorite machine learning package in Python scikitlearn makes doing advanced analysis in Python accessible to anyone At Yhat we make it easy to integrate these models into your production applications Thus eliminating the unnecessary dev time encountered productionizing analytical work

Greg Lamp Cofounder Yhat
20 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

1611 Rangespan

The Python scikitlearn toolkit is a core tool in the data science group at Rangespan. Its large collection of well documented models and algorithms allow our team of data scientists to prototype fast and quickly iterate to find the right solution to our learning problems. We find that scikitlearn is not only the right tool for prototyping but its careful and well tested implementation give us the confidence to run scikitlearn models in production.

Jurgen Van Gael, Data Science Director at Rangespan Ltd

1612 Birchbox

At Birchbox we face a range of machine learning problems typical to Ecommerce: product recommendation, user clustering, inventory prediction, trends detection, etc. Scikitlearn lets us experiment with many models, especially in the exploration phase of a new project. The data can be passed around in a consistent way, models are easy to save and reuse, updates keep us informed of new developments from the pattern discovery research community. Scikitlearn is an important tool for our team, built the right way in the right language.

Thierry Bertin, Mahieux, Birchbox Data Scientist

1613 Bestofmedia Group

Scikitlearn is our 1 toolkit for all things machine learning at Bestofmedia. We use it for a variety of tasks, eg. spam fighting, ad click prediction, various ranking models, thanks to the varied state-of-the-art algorithm implementations packaged into it. In the lab, it accelerates prototyping of complex pipelines. In production, I can say it has proven to be robust and efficient enough to be deployed for business critical components.

Eustache Diemert, Lead Scientist, Bestofmedia Group

16 Who is using scikitlearn 21

scikitlearn user guide Release 0213

1614 Changeorg

At changeorg we automate the use of scikitlearn’s RandomForestClassifier in our production systems to drive email targeting that reaches millions of users across the world each week In the lab scikitlearn’s easeofuse performance and overall variety of algorithms implemented has proved invaluable in giving us a single reliable source to turn to for our machinelearning needs

Vijay Ramesh Software Engineer in Datascience at Changeorg

1615 PHIMECA Engineering

At PHIMECA Engineering we use scikitlearn estimators as surrogates for expensivetoevaluate numerical models mostly but not exclusively finiteelement mechanical models for speeding up the intensive postprocessing operations involved in our simulationbased decision making framework Scikitlearn’s fitpredict API (Application Programming Interface) together with its efficient crossvalidation tools considerably eases the task of selecting the bestfit estimator We are also using scikitlearn for illustrating concepts in our training sessions Trainees are always impressed by the easeofuse of scikitlearn despite the apparent theoretical complexity of machine learning

Vincent Dubourg PHIMECA Engineering PhD Engineer

1616 HowAboutWe

At HowAboutWe scikitlearn lets us implement a wide array of machine learning techniques in analysis and in production despite having a small team We use scikitlearn’s classification algorithms to predict user behavior enabling us to for example estimate the value of leads from a given traffic source early in the lead’s tenure on our site Also our

22 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

users’ profiles consist of primarily unstructured data answers to openended questions so we use scikitlearn’s feature extraction and dimensionality reduction tools to translate these unstructured data into inputs for our matchmaking system

Daniel Weitzenfeld Senior Data Scientist at HowAboutWe
1617 PeerIndex

At PeerIndex we use scientific methodology to build the Influence Graph a unique dataset that allows us to identify who’s really influential and in which context To do this we have to tackle a range of machine learning and predictive modeling problems Scikitlearn has emerged as our primary tool for developing prototypes and making quick progress From predicting missing data and classifying tweets to clustering communities of social media users scikitlearn proved useful in a variety of applications Its very intuitive interface and excellent compatibility with other python tools makes it an indispensable tool in our daily research efforts

Ferenc Huszar Senior Data Scientist at Peerindex
1618 DataRobot

DataRobot is building next generation predictive analytics software to make data scientists more productive and scikitlearn is an integral part of our system The variety of machine learning techniques in combination with the solid implementations that scikitlearn offers makes it a onestopshopping library for machine learning in Python Moreover its consistent API (Application Programming Interface) welltested code and permissive licensing allow us to use it in a production environment Scikitlearn has literally saved us years of work we would have had to do ourselves to bring our product to market

Jeremy Achin CEO Cofounder DataRobot Inc

1619 OkCupid

We’re using scikitlearn at OkCupid to evaluate and improve our matchmaking system The range of features it has especially preprocessing utilities means we can use it for a wide variety of projects and it’s performant enough to handle the volume of data that we need to sort through The documentation is really thorough as well which makes the library quite easy to use

David Koh Senior Data Scientist at OkCupid

16 Who is using scikitlearn 23

scikitlearn user guide Release 0213

1620 Lovely

At Lovely we strive to deliver the best apartment marketplace with respect to our users and our listings From understanding user behavior improving data quality and detecting fraud scikitlearn is a regular tool for gathering insights predictive modeling and improving our product The easytoread documentation and intuitive architecture of the API (Application Programming Interface) makes machine learning both explorable and accessible to a wide range of python developers I’m con recommending that more developers and scientists try scikitlearn

Simon Frid Data Scientist Lead at Lovely

1621 Data Publica

Data Publica builds a new predictive sales tool for commercial and marketing teams called CRadar We extensively use scikitlearn to build segmentations of customers through clustering and to predict future customers based on past partnerships success or failure We also categorize companies using their website communication thanks to scikitlearn and its machine learning algorithm implementations Eventually machine learning makes it possible to detect weak signals that traditional tools cannot see All these complex tasks are performed in an easy and straightforward way thanks to the great quality of the scikitlearn framework

Guillaume Lebourgeois Samuel Charron Data Scientists at Data Publica

1622 Machinalis

Scikitlearn is the cornerstone of all the machine learning projects carried at Machinalis It has a consistent API (Application Programming Interface) wide selection of algorithms and lots of auxiliary tools to deal with the boilerplate We have used it in production environments on a variety of projects including clickthrough rate prediction information extraction and even counting sheep

In fact we use it so much that we’ve started to freeze our common use cases into Python packages some of them opensourced like FeatureForge Scikitlearn in one word Awesome

Rafael Carrascosa Lead developer

24 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

1623 solido

Scikitlearn is helping to drive Moore’s Law via Solido Solido creates computeraided design tools used by the majority of top20 semiconductor companies and fabs to design the bleedingedge chips inside smartphones auto mobiles and more Scikitlearn helps to power Solido’s algorithms for rareevent estimation worstcase verification optimization and more At Solido we are particularly fond of scikitlearn’s libraries for Gaussian Process models largescale regularized linear regression and classification Scikitlearn has increased our productivity because for many ML problems we no longer need to “roll our own” code This PyData 2014 talk has details

Trent McConaghy founder Solido Design Automation Inc

1624 INFONEA

We employ scikitlearn for rapid prototyping and custommade Data Science solutions within our inmemory based Business Intelligence Software INFONEA® As a welldocumented and comprehensive collection of stateoftheart algorithms and pipelining methods scikitlearn enables us to provide flexible and scalable scientific analysis solutions Thus scikitlearn is immensely valuable in realizing a powerful integration of Data Science technology within self service business analytics

Thorsten Kranz Data Scientist Coma Soft AG

1625 Dataiku

Our software Data Science Studio DSS enables users to create data services that combine ETL with Machine Learning Our Machine Learning module integrates many scikitlearn algorithms The scikitlearn library is a perfect integration with DSS because it offers algorithms for virtually all business cases Our goal is to offer a transparent and flexible tool that makes it easier to optimize time consuming aspects of building a data service preparing data and training machine learning algorithms on all types of data

Florian Douetteau CEO Dataiku

1626 Otto Group

Here at Otto Group one of global Big Five B2C online retailers we are using scikitlearn in all aspects of our daily work from data exploration to development of machine learning application to the productive deployment of those services It helps us to tackle machine learning problems ranging from ecommerce to logistics It consistent APIs enabled us to build the Palladium RESTAPI framework around it and continuously deliver scikitlearn based services

16 Who is using scikitlearn 25

scikitlearn user guide Release 0213

Christian Rammig Head of Data Science Otto Group

1627 Zopa

At Zopa the first ever PeertoPeer lending platform we extensively use scikitlearn to run the business and optimize our users’ experience It powers our Machine Learning models involved in credit risk fraud risk marketing and pricing and has been used for originating at least 1 billion GBP worth of Zopa loans It is very well documented powerful and simple to use We are grateful for the capabilities it has provided and for allowing us to deliver on our mission of making money simple and fair

Vlasios Vasileiou Head of Data Science Zopa

1628 MARS

ScikitLearn is integral to the Machine Learning Ecosystem at Mars Whether we’re designing better recipes for petfood or closely analysing our cocoa supply chain ScikitLearn is used as a tool for rapidly prototyping ideas and taking them to production This allows us to better understand and meet the needs of our consumers worldwide ScikitLearn’s featurerich toolset is easy to use and equips our associates with the capabilities they need to solve the business challenges they face every day

Michael Fitzke Next Generation Technologies Sr Leader Mars Inc

17 Release History

Release notes for current and recent releases are detailed on this page with previous releases linked below

Tip Subscribe to scikitlearn releases on librariesio to be notified when new versions are released

171 Legend for changelogs

- MAJOR FEATURE something big that you couldn’t do before
- FEATURE something that you couldn’t do before
- EFFICIENCY an existing feature now may not require as much computation or memory
- ENHANCEMENT a miscellaneous minor improvement
- FIX something that previously didn’t work as documented – or according to reasonable expectations – should now work
- API (Application Programming Interface) C HANGE you will need to change your code to have the same effect in the future or a feature will be removed in the future

26 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

18 Version 0213

July 30 2019

181 Changed models

The following estimators and functions when fit with the same data and parameters may produce different models from the previous version This often occurs due to changes in the modelling logic bug fixes or enhancements or in random sampling procedures

- The v0200 release notes failed to mention a backwards incompatibility in metricsmakescorer whenneedsprobaTrue andytrue is binary Now the scorer function is supposed to accept a 1D ypred ie probability of the positive class shape nsamples instead of a 2D ypred ie shape nsamples 2

182 Changelog

sklearncluster

- FIXFixed a bug in clusterKMeans where computation with initrandom was single threaded for njobs 1 ornjobs 1 12955 by Prabakaran Kumaresshan
- FIXFixed a bug in clusterOPTICS where users were unable to pass float minsamples and minclustersize 14496 by Fabian Klopfer and Hanmin Qin

sklearncompose

- FIXFixed an issue in composeColumnTransformer where using DataFrames whose column order differs between func fit and functransform could lead to silently passing incorrect columns to the remainder transformer 14237 by Andreas Schuderer

sklearndatasets

- FIXdatasetsfetchcaliforniahousing datasetsfetchcovtype datasets fetchkddcup99 datasetsfetcholivettifaces datasetsfetchrcv1 and datasetsfetchspeciesdistributions try to persist the previously cache using the new joblib if the cached data was persisted using the deprecated sklearnexternalsjoblib This behavior is set to be deprecated and removed in v023 14197 by Adrin Jalali

sklearnensemble

- FIX Fix zero division error in HistGradientBoostingClassifier and HistGradientBoostingRegressor 14024 by Nicolas Hug

sklearnimpute

- FIXFixed a bug in imputeSimpleImputer andimputeliterativeImputer so that no errors are thrown when there are missing values in training data 13974 by Frank Hoang

18 Version 0213 27

scikitlearn user guide Release 0213

sklearninspection

- FIXFixed a bug in inspectionplotpartialdependence wheretarget parameter was not being taken into account for multiclass problems 14393 by Guillem G Subies

sklearnlinearmodel

- FIXFixed a bug in linearmodelLogisticRegressionCV whererefitFalse would fail depending on the multiclass andpenalty parameters regression introduced in 021 14087 by Nicolas Hug
- FIXCompatibility fix for linearmodelARDRegression and Scipy130 Adapts to upstream changes to the default pinvh cutoff threshold which otherwise results in poor accuracy in some cases 14067 by Tim Staley

sklearnneighbors

- FIXFixed a bug in neighborsNeighborhoodComponentsAnalysis where the validation of initial parameters ncomponents maxiter andtol required too strict types 14092 by Jérémie du Boisber

sklearnranger

- FIXFixed bug in treeexporttext when the tree has one feature and a single feature name is passed in 14053 by Thomas Fan
- FIXFixed an issue with plottree where it displayed entropy calculations even for gini criterion in DecisionTreeClassifiers 13947 by Frank Hoang

19 Version 0212

24 May 2019

191 Changelog

sklearndecomposition

- FIXFixed a bug in crossdecompositionCCA improving numerical stability when Yis close to zero 13903 by Thomas Fan

sklearnmetrics

- FIXFixed a bug in metricspairwiseeuclideananddistances where a part of the distance matrix was left uninstantiated for sufficiently large float32 datasets regression introduced in 021 13910 by Jérémie du Boisberranger

28 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

sklearnpreprocessing

•FIXFixed a bug in preprocessingOneHotEncoder where the new drop parameter was not reflected

in getfeaturenames 13894 by James Myatt

sklearnutilssparsefuncs

•FIXFixed a bug where minmaxaxis would fail on 32bit systems for certain large inputs This

affects preprocessingMaxAbsScaler preprocessingnormalize and preprocessing

LabelBinarizer 13741 by Roddy MacSween

110 Version 0211

17 May 2019

This is a bugfix release to primarily resolve some packaging issues in version 0210 It also includes minor docu

mentation improvements and some bug fixes

1101 Changelog

sklearninspection

•FIXFixed a bug in inspectionpartialdependence to only check classifier and not regressor for

the multiclassmultioutput case 14309 by Guillaume Lemaitre

sklearnmetrics

•FIXFixed a bug in metricspairwiselocaldistances where it would raise AttributeError for

boolean metrics when X had a boolean dtype and Y None 13864 by Paresh Mathur

•FIXFixed two bugs in metricspairwiselocaldistances when njobs 1 First it used to return a

distance matrix with same dtype as input even for integer dtype Then the diagonal was not zeros for euclidean

metric when Y is X 13877 by Jérémie du Boisberranger

sklearnneighbors

•FIXFixed a bug in neighborsKernelDensity which could not be restored from a pickle if

sampleweight had been used 13772 by Aditya Vyas

111 Version 0210

May 2019

110 Version 0211 29

scikitlearn user guide Release 0213

1111 Changed models

The following estimators and functions when fit with the same data and parameters may produce different models from the previous version This often occurs due to changes in the modelling logic bug fixes or enhancements or in random sampling procedures

- discriminantanalysisLinearDiscriminantAnalysis for multiclass classification FIX
- discriminantanalysisLinearDiscriminantAnalysis with ‘eigen’ solver FIX
- linearmodelBayesianRidge FIX
- Decision trees and derived ensembles when both maxdepth andmaxleafnodes are set FIX
- linearmodelLogisticRegression andlinearmodelLogisticRegressionCV with ‘saga’ solver FIX
- ensembleGradientBoostingClassifier FIX
- sklearnfeatureextractiontextHashingVectorizer sklearnfeatureextractiontextTfidfVectorizer and sklearnfeatureextractiontextCountVectorizer FIX
- neuralnetworkMLPClassifier FIX
- svmSVCdecisionfunction and multiclassOneVsOneClassifier decisionfunction FIX
- linearmodelSGDClassifier and any derived classifiers FIX
- Any model using the linearmodelsagsagsolver function with a 0seed includ inglinearmodelLogisticRegression linearmodelLogisticRegressionCV linearmodelRidge andlinearmodelRidgeCV with ‘sag’ solver FIX
- linearmodelRidgeCV when using generalized crossvalidation with sparse inputs FIX

Details are listed in the changelog below

While we are trying to better inform users by providing this information we cannot assure that this list is complete

1112 Known Major Bugs

- The default maxiter forlinearmodelLogisticRegression is too small for many solvers given the defaulttol In particular we accidentally changed the default maxiter for the liblinear solver from 1000 to 100 iterations in 3591 released in version 016 In a future release we hope to choose better default maxiter andtol heuristically depending on the solver see 13317

1113 Changelog

Support for Python 34 and below has been officially dropped

sklearnbase

- API (Application Programming Interface) C HANGE The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutput

MultiOutputRegressor 13157 by Hanmin Qin

30 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

sklearncalibration

- ENHANCEMENT Added support to bin the data passed into calibrationcalibrationcurve by quantiles instead of uniformly between 0 and 1 13086 by Scott Cole

- ENHANCEMENT Allow ndimensional arrays as input for calibrationCalibratedClassifierCV 13485 by William de Vazelhes

sklearncluster

- MAJOR FEATURE A new clustering algorithm clusterOPTICS an algorithm related to cluster DBSCAN that has hyperparameters easier to set and that scales better by Shane Adrin Jalali Erich Schubert Hanmin Qin and Assia Benbihi

- FIXFixed a bug where clusterBirch could occasionally raise an AttributeError 13651 by Joel Nothman

- FIXFixed a bug in clusterKMeans where empty clusters weren't correctly relocated when using sample weights 13486 by Jérémie du Boisberranger

- API (Application Programming Interface) C HANGE Thencomponents attribute in clusterAgglomerativeClustering and clusterFeatureAgglomeration has been renamed to nconnectedcomponents 13427 by Stephane Couvreur

- ENHANCEMENT clusterAgglomerativeClustering andclusterFeatureAgglomeration now accept a distancethreshold parameter which can be used to find the clusters instead of nclusters 9069 by Vathsala Achar and Adrin Jalali

sklearncompose

- API (Application Programming Interface) C HANGE composeColumnTransformer is no longer an experimental feature 13835 by Hanmin Qin

sklearnndatasets

- FIXAdded support for 64bit group IDs and pointers in SVMLight files 10727 by Bryan K Woods

- FIXdatasetsloadsampleimages returns images with a deterministic order 13250 by Thomas Fan

sklearnndecomposition

- ENHANCEMENT decompositionKernelPCA now has deterministic output resolved sign ambiguity in eigenvalue decomposition of the kernel matrix 13241 by Aurélien Bellet

- FIXFixed a bug in decompositionKernelPCA fittransform now produces the correct output the same as fittransform in case of nonremoved zero eigenvalues removezeroeigFalse fitinversetransform was also accelerated by using the same trick asfittransform to compute the transform of X 12143 by Sylvain Marié

- FIXFixed a bug in decompositionNMF whereinit_nndsvd init_nndsvda and init_nndsvdar are allowed when ncomponents_nfeatures instead ofncomponents_minnsamples_nfeatures 11650 by Hossein Pourbozorg and Zijie ZJ Poh

scikitlearn user guide Release 0213

•API (Application Programming Interface) C HANGE The default value of the init argument in decomposition nonnegativefactorization will change from random toNone in version 023 to make it consistent with decompositionNMF A FutureWarning is raised when the default value is used 12988

by Zijie ZJ Poh

sklearnndiscriminantanalysis

•ENHANCEMENT discriminantanalysisLinearDiscriminantAnalysis now preserves float32 andfloat64 dtypes 8769 and 11000 by Thibault Sejourne

•FIXAChangedBehaviourWarning is now raised when discriminantanalysis

LinearDiscriminantAnalysis is given as parameter ncomponents minnfeatures

nclasses 1 andncomponents is changed to minnfeatures nclasses 1 if so

Previously the change was made but silently 11526 by William de Vazelhes

•FIXFixed a bug in discriminantanalysisLinearDiscriminantAnalysis where the pre

dicted probabilities would be incorrectly computed in the multiclass case 6848 by Agamemnon Krasoulis

andGuillaume Lemaitre

•FIXFixed a bug in discriminantanalysisLinearDiscriminantAnalysis where the pre

dicted probabilities would be incorrectly computed with eigen solver 11727 by Agamemnon Krasoulis

sklearndummy

•FIXFixed a bug in dummyDummyClassifier where thepredictproba method was returning int32

array instead of float64 for the stratified strategy 13266 by Christos Aridas

•FIXFixed a bug in dummyDummyClassifier where it was throwing a dimension mismatch error in

prediction time if a column vector ywithshapen 1 was given at fit time 13545 by Nick Sorros and

Adrin Jalali

sklearnensemble

•MAJOR FEATURE Add two new implementations of gradient boosting trees ensemble

HistGradientBoostingClassifier andensembleHistGradientBoostingRegressor

The implementation of these estimators is inspired by LightGBM and can be orders of magnitude faster than

ensembleGradientBoostingRegressor andensembleGradientBoostingClassifier

when the number of samples is larger than tens of thousands of samples The API (Application Programming Interface) of these new estimators

is slightly different and some of the features from ensembleGradientBoostingClassifier and

ensembleGradientBoostingRegressor are not yet supported

These new estimators are experimental which means that their results or their API (Application Programming Interface) might change without any

deprecation cycle To use them you need to explicitly import enablehistgradientboosting

explicitly require this experimental feature

from sklearnexperimental import enablehistgradientboosting noqa

now you can import normally from sklearnensemble

from sklearnensemble import HistGradientBoostingClassifier

12807 by Nicolas Hug

•FEATURE AddensembleVotingRegressor which provides an equivalent of ensemble

VotingClassifier for regression problems 12513 by Ramil Nugmanov and Mohamed Ali Jamaoui

32 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- EFFICIENCY MakeensembleIsolationForest prefer threads over processes when running with njobs 1 as the underlying decision tree fit calls do release the GIL This changes reduces memory usage and communication overhead 12543 by Isaac Storch and Olivier Grisel
- EFFICIENCY MakeensembleIsolationForest more memory efficient by avoiding keeping in memory each tree prediction 13260 by Nicolas Goix
- EFFICIENCY ensembleIsolationForest now uses chunks of data at prediction step thus capping the memory usage 13283 by Nicolas Goix
- EFFICIENCY sklearnensembleGradientBoostingClassifier andsklearnensemble GradientBoostingRegressor now keep the input yasfloat64 to avoid it being copied internally by trees 13524 by Adrin Jalali
- ENHANCEMENT Minimized the validation of X in ensembleAdaBoostClassifier andensemble AdaBoostRegressor 13174 by Christos Aridas
- ENHANCEMENT ensembleIsolationForest now exposes warmstart parameter allowing iterative addition of trees to an isolation forest 13496 by Peter Marko
- FIXThe values of featureimportances in all random forest based models ie ensembleRandomForestClassifier ensembleRandomForestRegressor ensembleExtraTreesClassifier ensembleExtraTreesRegressor ensemble RandomTreesEmbedding ensembleGradientBoostingClassifier andensemble GradientBoostingRegressor now -sum up to1 -all the single node trees in feature importance calculation are ignored -in case all trees have only one single node ie a root node feature importances will be an array of all zeros 13636 and 13620 by Adrin Jalali
- FIXFixed a bug in ensembleGradientBoostingClassifier andensemble GradientBoostingRegressor which didn't support scikitlearn estimators as the initial estimator Also added support of initial estimator which does not support sample weights 12436 by Jérémie du Boisberranger and 12983 by Nicolas Hug
- FIXFixed the output of the average path length computed in ensembleIsolationForest when the input is either 0 1 or 2 13251 by Albert Thomas and joshuakennethjones
- FIXFixed a bug in ensembleGradientBoostingClassifier where the gradients would be incor rectly computed in multiclass classification problems 12715 by Nicolas Hug
- FIXFixed a bug in ensembleGradientBoostingClassifier where validation sets for early stop ping were not sampled with stratification 13164 by Nicolas Hug
- FIXFixed a bug in ensembleGradientBoostingClassifier where the default initial prediction of a multiclass classifier would predict the classes priors instead of the log of the priors 12983 by Nicolas Hug
- FIXFixed a bug in ensembleRandomForestClassifier where thepredict method would error for multiclass multioutput forests models if any targets were strings 12834 by Elizabeth Sander
- FIXFixed a bug in ensemblegradientboostingLossFunction andensemble gradientboostingLeastSquaresError where the default value of learningrate in updateterminalregions is not consistent with the document and the caller functions Note however that directly using these loss functions is deprecated 6463 by movelikeriver
- FIXensemblepartialdependence and consequently the new version sklearninspection partialdependence now takes sample weights into account for the partial dependence computation when the gradient boosting model has been trained with sample weights 13193 by Samuel O Ronsin

scikitlearn user guide Release 0213

- API (Application Programming Interface) C HANGE ensemblepartialdependence andensembleplotpartialdependence are now deprecated in favor of inspectionpartialdependence andinspection plotpartialdependence 12599 by Trevor Stephens and Nicolas Hug
- FIXensembleVotingClassifier andensembleVotingRegressor were failing during fit in one of the estimators was set to None andsampleweight was notNone 13779 by Guillaume Lemaitre
- API (Application Programming Interface) C HANGE ensembleVotingClassifier andensembleVotingRegressor acceptdrop to disable an estimator in addition to None to be consistent with other estimators ie pipeline FeatureUnion andcomposeColumnTransformer 13780 by Guillaume Lemaitre

sklearnexternals

- API (Application Programming Interface) C HANGE Deprecated externalssix since we have dropped support for Python 27 12916 by Han min Qin

sklearnfeatureextraction

- FIXIfinputfile orinputfilename and a callable is given as the analyzer sklearn featureextractiontextHashingVectorizer sklearnfeatureextractiontext TfidfVectorizer andsklearnfeatureextractiontextCountVectorizer now read the data from the files and then pass it to the given analyzer instead of passing the file names or the file objects to the analyzer 13641 by Adrin Jalali

sklearnimpute

- MAJOR FEATURE AddedimputeIterativeImputer which is a strategy for imputing missing values by modeling each feature with missing values as a function of other features in a roundrobin fashion 8478 and 12177 by Sergey Feldman and Ben Lawson

The API (Application Programming Interface) of IterativeImputer is experimental and subject to change without any deprecation cycle To use them you need to explicitly import enableiterativeimputer

from sklearnexperimental import enableiterativeimputer noqa

now you can import normally from sklearnimpute

from sklearnimpute import IterativeImputer

- FEATURE TheimputeSimpleImputer andimputeIterativeImputer have a new parameter addindicator which simply stacks a imputeMissingIndicator transform into the output of the imputer's transform That allows a predictive estimator to account for missingness 12583 13601 by Danylo Baibak

- FIXInimputeMissingIndicator avoid implicit densification by raising an exception if input is sparse addmissingvalues property is set to 0 13240 by Bartosz Telenczuk

- FIXFixed two bugs in imputeMissingIndicator First when Xis sparse all the nonzero non missing values used to become explicit False in the transformed data Then when featuresmissingonly all features used to be kept if there were no missing values at all 13562 by Jérémie du Boisberranger

sklearninspection

new subpackage

34 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- FEATURE Partial dependence plots inspectionplotpartialdependence are now supported for any regressor or classifier provided that they have a predictproba method 12599 by Trevor Stephens and Nicolas Hug

sklearnisotonic

- FEATURE Allow different dtypes such as float32 in isotoniclsotonicRegression 8769 by Vlad Niculae

sklearnlinearmodel

- ENHANCEMENT linearmodelRidge now preserves float32 andfloat64 dtypes 8769 and 11000 by Guillaume Lemaitre and Joan Massich

- FEATURE linearmodelLogisticRegression and linearmodel

LogisticRegressionCV now support ElasticNet penalty with the ‘saga’ solver 11646 by Nicolas Hug

- FEATURE Addedlinearmodellarspathgram which is linearmodellarspath in the sufficient stats mode allowing users to compute linearmodellarspath without providing Xandy 11699 by Kuai Yu

- EFFICIENCY linearmodelmakedataset now preserves float32 andfloat64 dtypes reducing memory consumption in stochastic gradient SAG and SAGA solvers 8769 and 11000 by Nelle Varoquaux Arthur Imbert Guillaume Lemaitre and Joan Massich

- ENHANCEMENT linearmodelLogisticRegression now supports an unregularized objective when penaltynone is passed This is equivalent to setting Cnpinf with l2 regularization Not supported by the liblinear solver 12860 by Nicolas Hug

- ENHANCEMENT sparsecg solver inlinearmodelRidge now supports fitting the intercept ie fitinterceptTrue when inputs are sparse 13336 by Bartosz Telenczuk

- ENHANCEMENT The coordinate descent solver used in Lasso ElasticNet etc now issues a ConvergenceWarning when it completes without meeting the desired toleranbce 11754 and 13397 by Brent Fagan and Adrin Jalali

- FIXFixed a bug in linearmodelLogisticRegression andlinearmodel LogisticRegressionCV with ‘saga’ solver where the weights would not be correctly updated in some cases 11646 by Tom Dupre la Tour

- FIXFixed the posterior mean posterior covariance and returned regularization parameters in linearmodelBayesianRidge The posterior mean and the posterior covariance were not the ones computed with the last update of the regularization parameters and the returned regularization parameters were not the final ones Also fixed the formula of the log marginal likelihood used to compute the score when computescoreTrue 12174 by Albert Thomas

- FIXFixed a bug in linearmodelLassoLarsIC where user input copyXFalse at instance creation would be overridden by default parameter value copyXTrue in fit 12972 by Lucio Fernandez Arjona

- FIXFixed a bug in linearmodelLinearRegression that was not returning the same coefficients and intercepts with fitinterceptTrue in sparse and dense case 13279 by Alexandre Gramfort

- FIXFixed a bug in linearmodelHuberRegressor that was broken when Xwas of dtype bool 13328 by Alexandre Gramfort

111 Version 0210 35

scikitlearn user guide Release 0213

- FIXFixed a performance issue of saga andsag solvers when called in a joblibParallel setting with njobs 1 andbackendthreading causing them to perform worse than in the sequential case 13389 by Pierre Glaser
 - FIXFixed a bug in linearmodelstochasticgradientBaseSGDClassifier that was not deterministic when trained in a multiclass setting on several threads 13422 by Clément Doumouro
 - FIX Fixed bug in linearmodelridgeregression linearmodelRidge andlinearmodelRidgeClassifier that caused unhandled exception for arguments returninterceptTrue andsolverauto default or any other solver different from sag 13363 by Bartosz Telenczuk
 - FIXlinearmodelridgeregression will now raise an exception if returninterceptTrue and solver is different from sag Previously only warning was issued 13363 by Bartosz Telenczuk
 - FIXlinearmodelridgeregression will choose sparsecg solver for sparse inputs when solverauto andsampleweight is provided previously cholesky solver was selected 13363 by Bartosz Telenczuk
 - API (Application Programming Interface) C HANGE The use oflinearmodelarspath withXNone while passing Gram is deprecated in version 021 and will be removed in version 023 Use linearmodelarspathgram instead 11699 by Kuai Yu
 - API (Application Programming Interface) C HANGE linearmodellogisticregressionpath is deprecated in version 021 and will be removed in version 023 12821 by Nicolas Hug
 - FIXlinearmodelRidgeCV with generalized crossvalidation now correctly fits an intercept when fitinterceptTrue and the design matrix is sparse 13350 by Jérôme Dockès
 - sklearnmanifold
 - EFFICIENCY Makemanifoldtsnetrustworthiness use an inverted index instead of an npwhere lookup to find the rank of neighbors in the input space This improves efficiency in particular when computed with lots of neighbors andor small datasets 9907 by William de Vazelhes
 - sklearnmetrics
 - FEATURE Added themetricsmaxerror metric and a corresponding maxerror scorer for single output regression 12232 by Krishna Sangeeth
 - FEATURE Addmetricsmultilabelconfusionmatrix which calculates a confusion matrix with true positive false positive false negative and true negative counts for each class This facilitates the calculation of setwise metrics such as recall specificity fall out and miss rate 11179 by Shangwu Yao and Joel Nothman
 - FEATURE metricsjaccardscore has been added to calculate the Jaccard coefficient as an evaluation metric for binary multilabel and multiclass tasks with an interface analogous to metricsf1score 13151 by Gaurav Dhingra and Joel Nothman
 - FEATURE Addedmetricspairwiseaversinedistances which can be accessed with metricpairwise throughmetricspairwisedistances and estimators Haversine distance was previously available for nearest neighbors calculation 12568 by Wei Xue Emmanuel Arias and Joel Nothman
 - EFFICIENCY Fastermetricspairwisedistances with njobs 1 by using a threadbased backend instead of processbased backends 8216 by Pierre Glaser and Romuald Menuet
 - EFFICIENCY The pairwise manhattan distances with sparse input now uses the BLAS shipped with scipy instead of the bundled BLAS 12732 by Jérémie du Boisberranger
- 36 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- ENHANCEMENT Use label accuracy instead of microaverage onmetrics classificationreport to avoid confusion microaverage is only shown for multilabel or multiclass with a subset of classes because it is otherwise identical to accuracy 12334 by Emmanuel Arias Joel Nothman and Andreas Müller
 - ENHANCEMENT Addedbeta parameter to metricshomogeneitycompletenessvmeasure and metricsvmeasurescore to configure the tradeoff between homogeneity and completeness 13607 by Stephane Couvreur and Ivan Sanchez
 - FIXThe metric metricsr2score is degenerate with a single sample and now it returns NaN and raises exceptionsUndefinedMetricWarning 12855 by Pawel Sendyk
 - FIXFixed a bug where metricsbrierscoreloss will sometimes return incorrect result when there's only one class in ytrue 13628 by Hanmin Qin
 - FIXFixed a bug in metricslabelrankingaverageprecisionscore where sampleweight wasn't taken into account for samples with degenerate labels 13447 by Dan Ellis
 - API (Application Programming Interface) C HANGE The parameter labels inmetricshammingloss is deprecated in version 021 and will be removed in version 023 10580 by Reshama Shaikh and Sandra Mitrovic
 - FIXThe function metricspairwiseeuclidean distances and therefore several estimators with metricceclidean suffered from numerical precision issues with float32 features Precision has been increased at the cost of a small drop of performance 13554 by Celelibi and Jérémie du Boisberranger
 - API (Application Programming Interface) C HANGE metricsjaccardsimilarityscore is deprecated in favour of the more consistent metricsjaccardscore The former behavior for binary and multiclass targets is broken 13151 by Joel Nothman
- sklearnmixture
- FIXFixed a bug in mixtureBaseMixture and therefore on estimators based on it ie mixture GaussianMixture andmixtureBayesianGaussianMixture whererefitpredict andfit predict were not equivalent 13142 by Jérémie du Boisberranger
- sklearnmodelselection
- FEATURE ClassesGridSearchCV andRandomizedSearchCV now allow for refitcallable to add flexibility in identifying the best estimator See Balance model complexity and crossvalidated score 11354 by Wenhao Zhang Joel Nothman and Adrin Jalali
 - ENHANCEMENT ClassesGridSearchCV RandomizedSearchCV and methods crossvalscore crossvalpredict crossvalidate now print train scores when returntrainscores is True and verbose 2 Forlearningcurve andvalidationcurve only the latter is required 12613 and 12669 by Marc Torrellas
 - ENHANCEMENT Some CV (Cross-Validation) splitter classes and modelselectiontraintestsplit now raise ValueError when the resulting training set is empty 12861 by Nicolas Hug
 - FIXFixed a bug where modelselectionStratifiedKFold shuffles each class's samples with the samerandomstate makingshuffleTrue ineffective 13124 by Hanmin Qin
 - FIXAdded ability for modelselectioncrossvalpredict to handle multilabel and multioutputmulticlass targets with predictproba type methods 8773 by Stephen Hoover
 - FIXFixed an issue in crossvalpredict wheremethodpredictproba returned always 00 when one of the classes was excluded in a crossvalidation fold 13366 by Guillaume Fournier
- 111 Version 0210 37

scikitlearn user guide Release 0213

sklearnmulticlass

- FIXFixed an issue in multiclassOneVsOneClassifierdecisionfunction where the decisionfunction value of a given sample was different depending on whether the decisionfunction was evaluated on the sample alone or on a batch containing this same sample due to the scaling used in decisionfunction 10440 by Jonathan Ohayon

sklearnmultioutput

- FIXFixed a bug in multioutputMultiOutputClassifier where the predictproba method incorrectly checked for predictproba attribute in the estimator object 12222 by Rebekah Kim

sklearnneighbors

- MAJOR FEATURE AddedneighborsNeighborhoodComponentsAnalysis for metric learning which implements the Neighborhood Components Analysis algorithm 10058 by William de Vazelhes and John Chiotellis
- API (Application Programming Interface) C HANGE Methods in neighborsNearestNeighbors kneighbors radiusneighbors kneighborsgraph radiusneighborsgraph now raise NotFittedError rather than AttributeError when called before fit 12279 by Krishna Sangeeth

sklearnneuralnetwork

- FIXFixed a bug in neuralnetworkMLPClassifier andneuralnetworkMLPRegressor where the option shuffleFalse was being ignored 12582 by Sam Waterbury
- FIXFixed a bug in neuralnetworkMLPClassifier where validation sets for early stopping were not sampled with stratification In the multilabel case however splits are still not stratified 13164 by Nicolas Hug

sklearnpipeline

- FEATURE pipelinePipeline can now use indexing notation eg mypipeline01 to extract a subsequence of steps as another Pipeline instance A Pipeline can also be indexed directly to extract a particular step egmypipelinesvc rather than accessing namedsteps 2568 by Joel Nothman
- FEATURE Added optional parameter verbose inpipelinePipeline compose ColumnTransformer andpipelineFeatureUnion and corresponding make helpers for showing progress and timing of each step 11364 by Baze Petrushev Karan Desai Joel Nothman and Thomas Fan
- ENHANCEMENT pipelinePipeline now supports using passthrough as a transformer with the same effect as None 11144 by Thomas Fan
- ENHANCEMENT pipelinePipeline implements len and therefore lenpipeline returns the number of steps in the pipeline 13439 by Lakshya KD

sklearnpreprocessing

- FEATURE preprocessingOneHotEncoder now supports dropping one feature per category with a new drop parameter 12908 by Drew Johnston
- EFFICIENCY preprocessingOneHotEncoder andpreprocessingOrdinalEncoder now handle pandas DataFrames more efficiently 13253 by maikia

scikitlearn user guide Release 0213

- EFFICIENCY MakepreprocessingMultiLabelBinarizer cache class mappings instead of calculating it every time on the fly 12116 by Ekaterina Krivich and Joel Nothman
 - EFFICIENCY preprocessingPolynomialFeatures now supports compressed sparse row CSR (Compressed Sparse Row) matrices as input for degrees 2 and 3 This is typically much faster than the dense case as it scales with matrix density and expansion degree on the order of densitydegree and is much much faster than the compressed sparse column CSC case 12197 by Andrew Nystrom
 - EFFICIENCY Speed improvement in preprocessingPolynomialFeatures in the dense case Also added a new parameter order which controls output order for further speed performances 12251 by Tom Dupre la Tour
 - FIXFixed the calculation overflow when using a float16 dtype with preprocessingStandardScaler 13007 by Raffaello Baluyot
 - FIXFixed a bug in preprocessingQuantileTransformer andpreprocessingquantiletransform to force nquantiles to be at most equal to nsamples Values of nquantiles larger than nsamples were either useless or resulting in a wrong approximation of the cumulative distribution function estimator 13333 by Albert Thomas
 - API (Application Programming Interface) C HANGE The default value of copy inpreprocessingquantiletransform will change from False to True in 023 in order to make it more consistent with the default copy values of other functions in preprocessing and prevent unexpected side effects by modifying the value of Xinplace 13459 by Hunter McGushion
- sklearnsvm
- FIXFixed an issue in svmSVCdecisionfunction whendecisionfunctionshapeovr
- The decisionfunction value of a given sample was different depending on whether the decisionfunction was evaluated on the sample alone or on a batch containing this same sample due to the scaling used in decisionfunction 10440 by Jonathan Ohayon
- sklearnntree
- FEATURE Decision Trees can now be plotted with matplotlib using treeplottree without relying on the dot library removing a hardtoinstall dependency 8508 by Andreas Müller
 - FEATURE Decision Trees can now be exported in a human readable textual format using treeexporttext 6261 by Giuseppe Vettigli
 - FEATURE getnleaves andgetdepth have been added to treeBaseDecisionTree and consequently all estimators based on it including treeDecisionTreeClassifier treeDecisionTreeRegressor treeExtraTreeClassifier andtreeExtraTreeRegressor 12300 by Adrin Jalali
 - FIXTrees and forests did not previously predict multioutput classification targets with string labels despite accepting them in fit 11458 by Mitar Milutinovic
 - FIXFixed an issue with treeBaseDecisionTree and consequently all estimators based on it including treeDecisionTreeClassifier treeDecisionTreeRegressor treeExtraTreeClassifier andtreeExtraTreeRegressor where they used to exceed the given maxdepth by 1 while expanding the tree if maxleafnodes andmaxdepth were both specified by the user Please note that this also affects all ensemble methods using decision trees 12344 by Adrin Jalali
- 111 Version 0210 39

scikitlearn user guide Release 0213

- sklearnutils
- FEATURE utilsresample now accepts a stratify parameter for sampling according to class distributions 13549 by Nicolas Hug
 - API (Application Programming Interface) CHANGE Deprecated warnonddtype parameter from utilscheckarray andutilscheckXy Added explicit warning for dtype conversion in checkpairwisearrays if themetric being passed is a pairwise boolean metric 13382 by Prathmesh Savale
- Multiple modules
- MAJOR FEATURE Therepr method of all estimators used when calling printestimator has been entirely rewritten building on Python’s pretty printing standard library All parameters are printed by default but this can be altered with the printchangedonly option insklearnsetconfig 11705 by Nicolas Hug
 - MAJOR FEATURE Add estimators tags these are annotations of estimators that allow programmatic inspection of their capabilities such as sparse matrix support supported output types and supported methods Estimator tags also determine the tests that are run on an estimator when checkestimator is called Read more in theUser Guide 8022 by Andreas Müller
 - EFFICIENCY Memory copies are avoided when casting arrays to a different dtype in multiple estimators 11973 by Roman Yurchak
 - FIXFixed a bug in the implementation of the ourrandr helper function that was not behaving consistently across platforms 13422 by Madhura Parikh and Clément Doumouro
- Miscellaneous
- ENHANCEMENT Joblib is no longer vendored in scikitlearn and becomes a dependency Minimal supported version is joblib 011 however using version 013 is strongly recommended 13531 by Roman Yurchak
- 1114 Changes to estimator checks
- These changes mostly affect library developers
- Addcheckfitidempotent tocheckestimator which checks that when fitis called twice with the same data the ouput of predict predictproba transform and decisionfunction does not change 12328 by Nicolas Hug
 - Many checks can now be disabled or configured with Estimator Tags 8022 by Andreas Müller
- 1115 Code and Documentation Contributors
- Thanks to everyone who has contributed to the maintenance and improvement of the project since version 020 including
- adanhawth Aditya Vyas Adrin Jalali Agamemnon Krasoulis Albert Thomas Alberto Torres Alexandre Gramfort amourav Andrea Navarrete Andreas Mueller Andrew Nystrom assiaben Aurélien Bellet Bartosz Michałowski Bartosz Telenczuk bauks BenjaStudio bertrandhaut Bharat Raghunathan brentfagan Bryan Woods Cat Chenal Cheuk Ting Ho Chris Choe Christos Aridas Clément Doumouro Cole Smith Connossor Corey Levinson Dan Ellis Dan Stine Danylo Baibak datenkieker Denis Kataev Didi BarZev Dillon Gardner Dmitry Mottl Dmitry Vukolov Dougal J Sutherland Dowon drewmjohnton Dror Atariah Edward J Brown Ekaterina Krivich Eliza beth Sander Emmanuel Arias Eric Chang Eric Larson Erich Schubert esvhd Falak Feda Curic Federico Caselli
- 40 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

Frank Hoang Fbinse Xavier’ Finn O’Shea Gabriel Marzinotto Gabriel Vacaliuc Gabriele Calvo Gael Varoquaux
GauravAhlawat Giuseppe Vettigli Greg Gandenberger Guillaume Fournier Guillaume Lemaitre Gustavo De Mari
Pereira Hanmin Qin haroldfox hhuluqi Hunter McGushion Ian Sanders JackLangerman Jacopo Notarstefano
jakirkham James Bourbeau Jan Koch Jan S janvanrijn Jarrod Millman jdethurens jeremiedbb JF joaak Joan
Massich Joel Nothman Jonathan Ohayon Joris Van den Bossche josephsalmon Jérémie Méhault Katrin Leinwe
ber ken kms15 Koen Kossori Aruku Krishna Sangeeth Kuai Yu Kulbear Kushal Chauhan Kyle Jackson Lakshya
KD Leandro Hermida Lee Yi Jie Joel Lily Xiong Lisa Sarah Thomas Loic Esteve louib lukfa maikia mailliam
Manimaran Manuel LópezIbáñez Marc Torrellas Marco Gaido Marco Gorelli MarcoGorelli marineLM Mark
Hannel Martin Gubri Masstran mathurinm Matthew Roeschke Max Copeland melsyt mferrari3 Mickaël Schoent
gen Ming Li Mitar Mohammad Aftab Mohammed AbdelAal Mohammed Ibraheem Muhammad Hassaan Rafique
mwestt Naoya Iijima Nicholas Smith Nicolas Goix Nicolas Hug Nikolay Shebanov Oleksandr Pavlyk Oliver
Rausch Olivier Grisel Orestis Osman Owen Flanagan Paul Paczusi Pavel Soriano pavlos kallis Pawel Sendyk
peay Peter Peter Cock Peter Hausamann Peter Marko Pierre Glaser pierretallotte Pim de Haan Piotr Szyma ´nski
Prabakaran Kumaresshan Pradeep Reddy Raamana Prathmesh Savale Pulkit Maloo Quentin Batista Radostin Stoy
anov Raf Baluyot Rajdeep Dua Ramil Nugmanov Raúl García Calvo Rebekah Kim Reshama Shaikh Rohan
Lekhwani Rohan Singh Rohan Varma Rohit Kapoor Roman Feldbauer Roman Yurchak Romuald M Roopam
Sharma Ryan Rüdiger Busche Sam Waterbury Samuel O Ronsin SandroCasagrande Scott Cole Scott Lowe Se
bastian Raschka Shangwu Yao Shivam Kotwalia Shiyu Duan smarie Sriharsha Hatwar Stephen Hoover Stephen
Tierney Stéphane Couvreur surgan12 SylvainLan TakingItCasual Tashay Green thibsej Thomas Fan Thomas
J Fan Thomas Moreau Tom Dupré la Tour Tommy Tulio Casagrande Umar Farouk Umar Utkarsh Upadhyay
Vinayak Mehta Vishaal Kapoor Vivek Kumar Vlad Niculae vqean3 Wenhao Zhang William de Vazelhes xhan
Xing Han Lu xinyuli12 Yaroslav Halchenko Zach Griffith Zach Miller Zayd Hammoudeh Zhuyi Xue Zijie ZJ
Poh

112 Version 0204

July 30 2019

This is a bugfix release with some bug fixes applied to version 0203

1121 Changelog

The bundled version of joblib was upgraded from 0130 to 0132

sklearncluster

•FIXFixed a bug in clusterKMeans where KMeans initialisation could rarely result in an IndexError

11756 by Joel Nothman

sklearncompose

•FIXFixed an issue in composeColumnTransformer where using DataFrames whose column order
differs between func fit and funcstransform could lead to silently passing incorrect columns to the
remainder transformer 14237 by Andreas Schuderer

sklearnmodelselection

•FIXFixed a bug where modelselectionStratifiedKFold shuffles each class’s samples with the
samerandomstate makingshuffleTrue ineffective 13124 by Hanmin Qin

112 Version 0204 41

scikitlearn user guide Release 0.21.3

sklearn.neighbors

- FIX Fixed a bug in neighbors.KernelDensity which could not be restored from a pickle if sample_weight had been used 13772 by Aditya Vyas

1.13 Version 0.20.3

March 1, 2019

This is a bugfix release with some minor documentation improvements and enhancements to features released in 0.20.0

1.13.1 Changelog

sklearn.cluster

- FIX Fixed a bug in cluster.KMeans where computation was single threaded when n_jobs = 1 or n_jobs = -1 12949 by Prabakaran Kumareshan

sklearn.compose

- FIX Fixed a bug in compose.ColumnTransformer to handle negative indexes in the columns list of the transformers 12946 by Pierre Talbot

sklearn.covariance

- FIX Fixed a regression in covariance.graphical_lasso so that the case n_features > 2 is handled correctly 13276 by Aurélien Bellet

sklearn.decomposition

- FIX Fixed a bug in decomposition.sparse_encode where computation was single threaded when n_jobs = 1 or n_jobs = -1 13005 by Prabakaran Kumareshan

sklearn.datasets

- EFFICIENCY sklearn.datasets.fetch_openml now loads data by streaming avoiding high memory usage 13312 by Joris Van den Bossche

sklearn.feature_extraction

- FIX Fixed a bug in feature_extraction.text.CountVecorizer which would result in the sparse feature matrix having conflicting indices and indices precisions under very large vocabularies 11295 by Gabriel Vacaliuc

4.2 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

sklearnimpute

•FIXadd support for nonnumeric data in sklearnimputeMissingIndicator which was not supported while sklearnimputeSimpleImputer was supporting this for some imputation strategies 13046 by Guillaume Lemaitre

sklearnlinearmodel

•FIXFixed a bug in linearmodelMultiTaskElasticNet andlinearmodel

MultiTaskLasso which were breaking when warmstart True 12360 by Aakanksha Joshi sklearnpreprocessing

•FIXFixed a bug in preprocessingKBinsDiscretizer wherestrategykmeans fails with an error during transformation due to unsorted bin edges 13134 by Sandro Casagrande

•FIXFixed a bug in preprocessingOneHotEncoder where the deprecation of categoricalfeatures was handled incorrectly in combination with handleunknownignore 12881 by Joris Van den Bossche

•FIXBins whose width are too small ie 1e8 are removed with a warning in preprocessing KBinsDiscretizer 13165 by Hanmin Qin

sklearnsvm

•FIXFixed a bug in svmSVC svmNuSVC svmSVR svmNuSVR andsvmOneClassSVM where the scale option of parameter gamma is erroneously defined as 1 / nfeatures Xstd It's now defined as1 / nfeatures Xvar 13221 by Hanmin Qin

1132 Code and Documentation Contributors

With thanks to

Adrin Jalali Agamemnon Krasoulis Albert Thomas Andreas Mueller Aurélien Bellet bertrandhaut Bharat Raghu nathan Dowon Emmanuel Arias Fbinse Xavier Finn O'Shea Gabriel Vacaliuc Gael Varoquaux Guillaume Lemaitre Hanmin Qin joaak Joel Nothman Joris Van den Bossche Jérémie Méhault kms15 Kossori Aruku Lak shya KD maikia Manuel LópezIbáñez Marco Gorelli MarcoGorelli mferrari3 Mickaël Schoentgen Nicolas Hug pavlos kallis Pierre Glaser pierretallotte Prabakaran Kumaresshan Reshama Shaikh Rohit Kapoor Roman Yurchak SandroCasagrande Tashay Green Thomas Fan Vishaal Kapoor Zhuyi Xue Zijie ZJ Poh

114 Version 0202

December 20 2018

This is a bugfix release with some minor documentation improvements and enhancements to features released in 0200

114 Version 0202 43

scikitlearn user guide Release 0213

1141 Changed models

The following estimators and functions when fit with the same data and parameters may produce different models from the previous version This often occurs due to changes in the modelling logic bug fixes or enhancements or in random sampling procedures

- sklearnneighbors whenmetricjaccard bug fix
- use ofseuclidean ormahalanobis metrics in some cases bug fix

1142 Changelog

sklearncompose

- FIXFixed an issue in composemakecolumntransformer which raises unexpected error when columns is pandas Index or pandas Series 12704 by Hanmin Qin

sklearnmetrics

- FIX Fixed a bug in metricspairwiselocaldistances andmetrics

pairwiselocaldistanceschunked where parameters Vofseuclidean andVlofmahalanobis

metrics were computed after the data was split into chunks instead of being precomputed on whole data

12701 by Jeremie du Boisberranger

sklearnneighbors

- FIXFixedsklearnneighborsDistanceMetric jaccard distance function to return 0 when two all zero vectors are compared 12685 by Thomas Fan

sklearnutils

- FIXCallingutilscheckarray onpandasSeries with categorical data which raised an error in 0200 now returns the expected output again 12699 by Joris Van den Bossche

1143 Code and Documentation Contributors

With thanks to

adanhawth Adrin Jalali Albert Thomas Andreas Mueller Dan Stine Feda Curic Hanmin Qin Jan S jeremiedbb

Joel Nothman Joris Van den Bossche josephsalmon Katrin Leinweber Loic Esteve Muhammad Hassaan Rafique

Nicolas Hug Olivier Grisel Paul Paczuski Reshama Shaikh Sam Waterbury Shivam Kotwalia Thomas Fan

115 Version 0201

November 21 2018

This is a bugfix release with some minor documentation improvements and enhancements to features released in

0200 Note that we also include some API (Application Programming Interface) changes in this release so you might get some extra warnings after updating from 0200 to 0201

44 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

1151 Changed models

The following estimators and functions when fit with the same data and parameters may produce different models from the previous version This often occurs due to changes in the modelling logic bug fixes or enhancements or in random sampling procedures

- decompositionIncrementalPCA bug fix

1152 Changelog

sklearncluster

- EFFICIENCY makeclusterMeanShift no longer try to do nested parallelism as the overhead would hurt performance significantly when njobs 1 12159 by Olivier Grisel

- FIXFixed a bug in clusterDBSCAN with precomputed sparse neighbors graph which would add explicitly zeros on the diagonal even when already present 12105 by Tom Dupre la Tour

sklearncompose

- FIXFixed an issue in composeColumnTransformer when stacking columns with types not convertible to a numeric 11912 by Adrin Jalali

- API (Application Programming Interface) C HANGE composeColumnTransformer now applies the sparsethreshold even if all transformation results are sparse 12304 by Andreas Müller

- API (Application Programming Interface) C HANGE composemakecolumntransformer now expects transformer columns instead ofcolumns transformer to keep consistent with composeColumnTransformer 12339

by Adrin Jalali

sklearndatasets

- FIXdatasetsfetchopenml to correctly use the local cache 12246 by Jan N van Rijn

- FIXdatasetsfetchopenml to correctly handle ignore attributes and row id attributes 12330 by Jan N van Rijn

- FIXFixed integer overflow in datasetsmakeclassification for values of ninformative parameter larger than 64 10811 by Roman Feldbauer

- FIXFixed olivetti faces dataset DESCR attribute to point to the right location in datasets fetcholivettifaces 12441 by Jérémie du Boisberranger

- FIXdatasetsfetchopenml to retry downloading when reading from local cache fails 12517 by Thomas Fan

sklearncomposition

- FIXFixed a regression in decompositionIncrementalPCA where 0200 raised an error if the number of samples in the final batch for fitting IncrementalPCA was smaller than ncomponents 12234 by Ming Li

scikitlearn user guide Release 0213

sklearnensemble

- FIX Fixed a bug mostly affecting ensembleRandomForestClassifier where classweightbalancedsubsample failed with more than 32 classes 12165 by Joel Nothman
- FIX Fixed a bug affecting ensembleBaggingClassifier ensembleBaggingRegressor and ensembleIsolationForest where maxfeatures was sometimes rounded down to zero 12388

by Connor Tann

sklearnfeatureextraction

- FIX Fixed a regression in v0200 where featureextractiontextCountVectorizer and other text vectorizers could error during stop words validation with custom preprocessors or tokenizers 12393 by Roman Yurchak

sklearnlinearmodel

- FIX linearmodelSGDClassifier and variants with earlystoppingTrue would not use a consistent validation split in the multiclass case and this would cause a crash when using those estimators as part of parallel parameter search or crossvalidation 12122 by Olivier Grisel
- FIX Fixed a bug affecting SGDClassifier in the multiclass case Each oneversusall step is run in a joblibParallel call and mutating a common parameter causing a segmentation fault if called within a backend using processes and not threads We now use requiresharedmem at thejoblibParallel instance creation 12518 by Pierre Glaser and Olivier Grisel

sklearnmetrics

- FIX Fixed a bug in metricspairwisepairwisedistancesargminmin which returned the square root of the distance when the metric parameter was set to “euclidean” 12481 by Jérémie du Boisber

ranger

- FIX Fixed a bug in metricspairwisepairwisedistanceschunked which didn’t ensure the diagonal is zero for euclidean distances 12612 by Andreas Müller
- API (Application Programming Interface) C HANGE Themetricscalinskiharabaszscore has been renamed to metricscalinskiharabaszscore and will be removed in version 023 12211 by Lisa Thomas Mark Hannel

and Melissa Ferrari

sklearnmixture

- FIX Ensure that the fitpredict method of mixtureGaussianMixture andmixtureBayesianGaussianMixture always yield assignments consistent with fit followed by predict even if the convergence criterion is too loose or not met 12451 by Olivier Grisel

sklearnneighbors

- FIX force the parallelism backend to threading forneighborsKDTree andneighborsBallTree in Python 27 to avoid pickling errors caused by the serialization of their methods 12171 by Thomas Moreau

46 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

sklearnpreprocessing

- FIXFixed bug in preprocessingOrdinalEncoder when passing manually specified categories 12365 by Joris Van den Bossche
 - FIXFixed bug in preprocessingKBinsDiscretizer where thetransform method mutates the encoder attribute The transform method is now thread safe 12514 by Hanmin Qin
 - FIXFixed a bug in preprocessingPowerTransformer where the YeoJohnson transform was incorrect for lambda parameters outside of 0 2 12522 by Nicolas Hug
 - FIXFixed a bug in preprocessingOneHotEncoder where transform failed when set to ignore unknown numpy strings of different lengths 12471 by Gabriel Marzinotto
 - API (Application Programming Interface) C HANGE The default value of the method argument in preprocessingpowertransform will be changed from boxcox toyeojohnson to matchpreprocessingPowerTransformer in version 023 A FutureWarning is raised when the default value is used 12317 by Eric Chang
- sklearnutils
- FIXUse float64 for mean accumulator to avoid floating point precision issues in preprocessing StandardScaler anddecompositionIncrementalPCA when using float32 datasets 12338 by bauks
 - FIXCallingutilscheckarray onpandasSeries which raised an error in 0200 now returns the expected output again 12625 by Andreas Müller

Miscellaneous

- FIXWhen using site joblib by setting the environment variable SKLEARNSITEJOBLIB added compatibility with joblib 011 in addition to 012 12350 by Joel Nothman and Roman Yurchak
 - FIXMake sure to avoid raising FutureWarning when calling npvstack with numpy 116 and later use list comprehensions instead of generator expressions in many locations of the scikitlearn code base 12467 by Olivier Grisel
 - API (Application Programming Interface) C HANGE Removed all mentions of sklearnexternalsjoblib and deprecated joblib methods exposed in sklearnutils except for utilsparallelbackend andutils registerparallelbackend which allow users to configure parallel computation in scikitlearn Other functionalities are part of joblib package and should be used directly by installing it The goal of this change is to prepare for unvendoring joblib in future version of scikitlearn 12345 by Thomas Moreau
- 1153 Code and Documentation Contributors

With thanks to

Adrin Jalali Andrea Navarrete Andreas Mueller bauks BenjaStudio Cheuk Ting Ho Connossor Corey Levin son Dan Stine datenkieker Denis Kataev Dillon Gardner Dmitry Vukolov Dougal J Sutherland Edward J Brown Eric Chang Federico Caselli Gabriel Marzinotto Gael Varoquaux GauravAhlawat Gustavo De Mari Pereira Han min Qin haroldfox JackLangerman Jacopo Notarstefano janvanrijn jdethurens jeremiedbb Joel Nothman Joris Van den Bossche Koen Kushal Chauhan Lee Yi Jie Joel Lily Xiong mailliam Mark Hannel melsyt Ming Li Nicholas Smith Nicolas Hug Nikolay Shebanov Oleksandr Pavlyk Olivier Grisel Peter Hausamann Pierre Glaser Pulkit Maloo Quentin Batista Radostin Stoyanov Ramil Nugmanov Rebekah Kim Reshama Shaikh Rohan Singh Roman Feldbauer Roman Yurchak Roopam Sharma Sam Waterbury Scott Lowe Sebastian Raschka Stephen Tier ney SylvainLan TakingItCasual Thomas Fan Thomas Moreau Tom Dupré la Tour Tulio Casagrande Utkarsh Upadhyay Xing Han Lu Yaroslav Halchenko Zach Miller

115 Version 0201 47

scikitlearn user guide Release 0213

116 Version 0200

September 25 2018

This release packs in a mountain of bug fixes features and enhancements for the Scikitlearn library and improvements to the documentation and examples Thanks to our contributors

This release is dedicated to the memory of Raghav Rajagopalan

Warning Version 020 is the last version of scikitlearn to support Python 27 and Python 34 Scikitlearn 021 will require Python 35 or higher

1161 Highlights

We have tried to improve our support for common datascience usecases including missing values categorical variables heterogeneous data and featurerepresentations with unusual distributions Missing values in features represented by NaNs are now accepted in columnwise preprocessing such as scalers Each feature is fitted disregarding NaNs and data containing NaNs can be transformed The new impute module provides estimators for learning despite missing data

ColumnTransformer handles the case where different features or columns of a pandasDataFrame need different preprocessing String or pandas Categorical columns can now be encoded with OneHotEncoder or OrdinalEncoder

TransformedTargetRegressor helps when the regression target needs to be transformed to be modeled

PowerTransformer andKBinsDiscretizer joinQuantileTransformer as nonlinear transformations

Beyond this we have added sampleweight support to several estimators including KMeans

BayesianRidge andKernelDensity and improved stopping criteria in others including MLPRegressor

GradientBoostingRegressor andSGDRegressor

This release is also the first to be accompanied by a Glossary of Common Terms and API (Application Programming Interface) Elements developed by Nothman The glossary is a reference resource to help users and contributors become familiar with the terminology and conventions used in Scikitlearn

Sorry if your contribution didn't make it into the highlights There's a lot here

1162 Changed models

The following estimators and functions when fit with the same data and parameters may produce different models from the previous version This often occurs due to changes in the modelling logic bug fixes or enhancements or in random sampling procedures

- clusterMeanShift bug fix
 - decompositionIncrementalPCA in Python 2 bug fix
 - decompositionSparsePCA bug fix
 - ensembleGradientBoostingClassifier bug fix affecting feature importances
 - isotonicIsotonicRegression bug fix
 - linearmodelARDRegression bug fix
 - linearmodelLogisticRegressionCV bug fix
 - linearmodelOrthogonalMatchingPursuit bug fix
- 48 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- linearmodelPassiveAggressiveClassifier bug fix
 - linearmodelPassiveAggressiveRegressor bug fix
 - linearmodelPerceptron bug fix
 - linearmodelSGDClassifier bug fix
 - linearmodelSGDRegressor bug fix
 - metricsrocaucscore bug fix
 - metricsroccurve bug fix
 - neuralnetworkBaseMultilayerPerceptron bug fix
 - neuralnetworkMLPClassifier bug fix
 - neuralnetworkMLPRegressor bug fix
 - The v0190 release notes failed to mention a backwards incompatibility with modelselection
- StratifiedKFold when shuffle=True due to 7823

Details are listed in the changelog below

While we are trying to better inform users by providing this information we cannot assure that this list is complete

1163 Known Major Bugs

- 11924 linearmodelLogisticRegressionCV with solver lbfgs and multiclassmultinomial may be nondeterministic or otherwise broken on macOS This appears to be the case on Travis CI servers but has not been confirmed on personal MacBooks This issue has been present in previous releases
- 9354 metrics pairwise euclidean distances which is used several times throughout the library gives results with poor precision which particularly affects its use with 32bit float inputs This became more problematic in versions 018 and 019 when some algorithms were changed to avoid casting 32bit data into 64bit

1164 Changelog

Support for Python 3.3 has been officially dropped

sklearn.cluster

- MAJOR FEATURE clusterAgglomerativeClustering now supports Single Linkage clustering via linkage='single' 9372 by Leland McInnes and Steve Astels
- FEATURE clusterKMeans and clusterMiniBatchKMeans now support sample weights via new parameter sample_weight in fit function 10933 by Johannes Hansen
- EFFICIENCY clusterKMeans clusterMiniBatchKMeans and clusterkmeans passed with algorithm='full' now enforces row-major ordering improving runtime 10471 by Gaurav Dhingra
- EFFICIENCY clusterDBSCAN now is parallelized according to n_jobs regardless of algorithm 8003 by Joël Billaud
- ENHANCEMENT clusterKMeans now gives a warning if the number of distinct clusters found is smaller than n_clusters This may occur when the number of distinct points in the data set is actually smaller than the number of clusters one is looking for 10059 by Christian Braune

116 Version 0200 49

scikitlearn user guide Release 0213

- FIXFixed a bug where the fit method ofclusterAffinityPropagation stored cluster centers as 3d array instead of 2d array in case of nonconvergence For the same class fixed undefined and arbitrary behavior in case of training data where all samples had equal similarity 9612 By Jonatan Samoocha
 - FIXFixed a bug in clusterspectralclustering where the normalization of the spectrum was using a division instead of a multiplication 8129 by Jan Margeta Guillaume Lemaitre and Devansh D
 - FIXFixed a bug in clusterkmeanselkan where the returned iteration was 1 less than the correct value Also added the missing niter attribute in the docstring of clusterKMeans 11353 by Jeremie du Boisberranger
 - FIXFixed a bug in clustermeanshift where the assigned labels were not deterministic if there were multiple clusters with the same intensities 11901 by Adrin Jalali
 - API (Application Programming Interface) C HANGE Deprecate poolingfunc unused parameter in cluster AgglomerativeClustering 9875 by Kumar Ashutosh
- sklearncompose
- New module
 - MAJOR FEATURE AddedcomposeColumnTransformer which allows to apply different transformers to different columns of arrays or pandas DataFrames 9012 by Andreas Müller and Joris Van den Bossche and 11315 by Thomas Fan
 - MAJOR FEATURE Added thecomposeTransformedTargetRegressor which transforms the target y before fitting a regression model The predictions are mapped back to the original space via an inverse transform 9041 by Andreas Müller and Guillaume Lemaitre
- sklearncovariance
- EFFICIENCY Runtime improvements to covarianceGraphicalLasso 9858 by Steven Brown
 - API (Application Programming Interface) C HANGE Thecovariancegraphlasso covarianceGraphLasso andcovariance GraphLassoCV have been renamed to covariancegraphicallasso covariance GraphicalLasso andcovarianceGraphicalLassoCV respectively and will be removed in version 022 9993 by Artiem Krinitsyn
- sklearndatasets
- MAJOR FEATURE Addeddatasetsfetchopenml to fetch datasets from OpenML OpenML is a free open data sharing platform and will be used instead of mldata as it provides better service availability 9908 by Andreas Müller and Jan N van Rijn
 - FEATURE Indatasetsmakeblobs one can now pass a list to the nsamples parameter to indicate the number of samples to generate per cluster 8617 by Maskani Filali Mohamed and Konstantinos Katrioplas
 - FEATURE Addfilename attribute to datasets that have a CSV file 9101 by alex33 and Maskani Filali Mohamed
 - FEATURE returnXy parameter has been added to several dataset loaders 10774 by Chris Catalfo
 - FIXFixed a bug in datasetsloadboston which had a wrong data point 10795 by Takeshi Yoshizawa
 - FIXFixed a bug in datasetsloadiris which had two wrong data points 11082 by Sadhana Srini vasan and Hanmin Qin
- 50 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- FIXFixed a bug in datasetsfetchkddcup99 where data were not properly shuffled 9731 by Nicolas Goix
 - FIXFixed a bug in datasetsmakecircles where no odd number of data points could be generated 10045 by Christian Braune
 - API (Application Programming Interface) CHANGE Deprecated sklearn.datasets.fetch_ml_data to be removed in version 0.22 ml data.org is no longer operational Until removal it will remain possible to load cached datasets 11466 by Joel Nothman
 - sklearn.decomposition
 - FEATURE decompositiondictlearning functions and models now support positivity constraints This applies to the dictionary and sparse code 6374 by John Kirkham
 - FEATURE FIXdecompositionSparsePCA now exposes normalize_components When set to True the train and test data are centered with the train mean respectively during the fit phase and the transform phase This fixes the behavior of SparsePCA When set to False which is the default the previous abnormal behaviour still holds The False value is for backward compatibility and should not be used 11585 by Ivan Panico
 - EFFICIENCY Efficiency improvements in decompositiondictlearning 11420 and others by John Kirkham
 - FIXFix for uninformative error in decompositionIncrementalPCA now an error is raised if the number of components is larger than the chosen batch size The n_components=None case was adapted accordingly 6452 By Wally Gauze
 - FIXFixed a bug where the partial_fit method of decompositionIncrementalPCA used integer division instead of float division on Python 2 9492 by James Bourbeau
 - FIXIn decompositionPCA selecting a n_components parameter greater than the number of samples now raises an error Similarly the n_components=None case now selects the minimum of n_samples and n_features 8484 by Wally Gauze
 - FIXFixed a bug in decompositionPCA where users will get unexpected error with large datasets when n_components=ml on Python 3 versions 9886 by Hanmin Qin
 - FIXFixed an underflow in calculating KLdivergence for decompositionNMF 10142 by Tom Dupre la Tour
 - FIXFixed a bug in decompositionSparseCoder when running OMP sparse coding in parallel using read-only memory mapped data structures 5956 by Vignesh Birodkar and Olivier Grisel
 - sklearn.discriminantanalysis
 - EFFICIENCY Memory usage improvement for class_means and class_cov in discriminantanalysis 10898 by Nanxin Chen
 - sklearn.dummy
 - FEATURE dummyDummyRegressor now has a return_std option in its predict method The returned standard deviations will be zeros
 - FEATURE dummyDummyClassifier and dummyDummyRegressor now only require X to be an object with finite length or shape 9832 by Vrishank Bhardwaj
- 116 Version 0.20.51

scikitlearn user guide Release 0213

- FEATURE dummyDummyClassifier anddummyDummyRegressor can now be scored without sup
plying test samples 11951 by Rüdiger Busche
sklearnensemble
 - FEATURE ensembleBaggingRegressor andensembleBaggingClassifier can now be fit
with missingnonfinite values in X andor multioutput Y to support wrapping pipelines that perform their
own imputation 9707 by Jimmy Wan
 - FEATURE ensembleGradientBoostingClassifier and ensemble
GradientBoostingRegressor now support early stopping via niternochange
validationfraction andtol 7071 by Raghav RV
 - FEATURE Addednamedestimators parameter in ensembleVotingClassifier to access fitted
estimators 9157 by Herilalaina Rakotoarison
 - FIXFixed a bug when fitting ensembleGradientBoostingClassifier orensemble
GradientBoostingRegressor withwarmstartTrue which previously raised a segmentation fault
due to a nonconversion of CSC matrix into CSR (Compressed Sparse Row) format expected by decisionfunction Similarly
Fortranordered arrays are converted to Cordered arrays in the dense case 9991 by Guillaume Lemaitre
 - FIXFixed a bug in ensembleGradientBoostingRegressor andensemble
GradientBoostingClassifier to have feature importances summed and then normalized rather
than normalizing on a pertree basis The previous behavior overweighted the Gini importance of features that
appear in later stages This issue only affected feature importances 11176 by Gil Forsyth
 - API (Application Programming Interface) C HANGE The default value of the nestimators parameter of ensemble
RandomForestClassifier ensembleRandomForestRegressor ensemble
ExtraTreesClassifier ensembleExtraTreesRegressor and ensemble
RandomTreesEmbedding will change from 10 in version 020 to 100 in 022 A FutureWarning is
raised when the default value is used 11542 by Anna Ayzenshtat
 - API (Application Programming Interface) C HANGE Classes derived from ensembleBaseBagging The attribute estimatorssamples
will return a list of arrays containing the indices selected for each bootstrap instead of a list of arrays containing
the mask of the samples selected for each bootstrap Indices allows to repeat samples while mask does not allow
this functionality 9524 by Guillaume Lemaitre
 - FIXensembleBaseBagging where one could not deterministically reproduce fit result using the object
attributes when randomstate is set 9723 by Guillaume Lemaitre
sklearnfeatureextraction
 - FEATURE Enable the call to getfeaturenames in unfitted featureextractiontext
CountVectorizer initialized with a vocabulary 10908 by Mohamed Maskani
 - ENHANCEMENT idf can now be set on a featureextractiontextTfidfTransformer
10899 by Sergey Melderis
 - FIXFixed a bug in featureextractionimageextractpatches2d which would throw an
exception if maxpatches was greater than or equal to the number of all possible patches rather than simply
returning the number of possible patches 10101 by Varun Agrawal
 - FIXFixed a bug in featureextractiontextCountVectorizer featureextraction
textTfidfVectorizer featureextractiontextHashingVectorizer to support 64 bit
sparse array indexing necessary to process large datasets with more than $2 \cdot 10^9$ tokens words or ngrams
9147 by ClaesFredrik Mannby and Roman Yurchak
- 52 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- FIXFixed bug in featureextractiontextTfidfVectorizer which was ignoring the parameterdtype In addition featureextractiontextTfidfTransformer will preserve dtype for floating and raise a warning if dtype requested is integer 10441 by Mayur Kulkarni and Guillaume Lemaitre
 - sklearnfeatureselection
 - FEATURE Added select K best features functionality to featureselectionSelectFromModel 6689 by Nihar Sheth and Quazi Rahman
 - FEATURE Addedminfeaturestoselect parameter to featureselectionRFECV to bound evaluated features counts 11293 by Brent Yi
 - FEATURE featureselectionRFECV 's fit method now supports groups 9656 by Adam Greenhall
 - FIXFixed computation of nfeaturestocompute for edge case with tied CV (Cross-Validation) scores in featureselectionRFECV 9222 by Nick Hoh
 - sklearngaussianprocess
 - EFFICIENCY IngaussianprocessGaussianProcessRegressor methodpredict is faster when using returnstdTrue in particular more when called several times in a row 9234 by andrewww and Minghui Liu
 - sklearnimpute
 - New module adopting preprocessingImputer asimputeSimpleImputer with minor changes see under preprocessing below
 - MAJOR FEATURE AddedimputeMissingIndicator which generates a binary indicator for missing values 8075 by Maniteja Nandana and Guillaume Lemaitre
 - FEATURE TheimputeSimpleImputer has a new strategy constant to complete missing values with a fixed one given by the fillvalue parameter This strategy supports numeric and nonnumeric data and so does the mostfrequent strategy now 11211 by Jeremie du Boisberranger
 - sklearnisotonic
 - FIXFixed a bug in isotonicIsotonicRegression which incorrectly combined weights when fitting a model to data involving points with identical X values 9484 by Dallas Card
 - sklearnlinearmodel
 - FEATURE linearmodelSGDClassifier linearmodelSGDRegressor linearmodelPassiveAggressiveClassifier linearmodelPassiveAggressiveRegressor and linearmodelPerceptron now expose earlystopping validationfraction and niternochange parameters to stop optimization monitoring the score on a validation set A new learning rateadaptive strategy divides the learning rate by 5 each time niternochange consecutive epochs fail to improve the model 9043 by Tom Dupre la Tour
 - FEATURE Add sampleweight parameter to the fit method of linearmodelBayesianRidge for weighted linear regression 10112 by Peter St John
- 116 Version 0200 53

scikitlearn user guide Release 0213

- FIXFixed a bug in logisticlogisticregressionpath to ensure that the returned coefficients are correct when multiclassmultinomial. Previously some of the coefficients would override each other leading to incorrect results in linearmodelLogisticRegressionCV 11724 by Nicolas Hug
 - FIXFixed a bug in linearmodelLogisticRegression where when using the parameter multiclassmultinomial thepredictproba method was returning incorrect probabilities in the case of binary outcomes 9939 by Roger Westover
 - FIXFixed a bug in linearmodelLogisticRegressionCV where thescore method always computes accuracy not the metric given by the scoring parameter 10998 by Thomas Fan
 - FIXFixed a bug in linearmodelLogisticRegressionCV where the ‘ovr’ strategy was always used to compute crossvalidation scores in the multiclass setting even if multinomial was set 8720 by William de Vazelhes
 - FIXFixed a bug in linearmodelOrthogonalMatchingPursuit that was broken when setting normalizeFalse 10071 by Alexandre Gramfort
 - FIXFixed a bug in linearmodelARDRegression which caused incorrectly updated estimates for the standard deviation and the coefficients 10153 by Jörg Döpfert
 - FIXFixed a bug in linearmodelARDRegression andlinearmodelBayesianRidge which caused NaN predictions when fitted with a constant target 10095 by Jörg Döpfert
 - FIXFixed a bug in linearmodelRidgeClassifierCV where the parameter storecvvalues was not implemented though it was documented in cvvalues as a way to set up the storage of cross validation values for different alphas 10297 by Mabel VillalbalJiménez
 - FIXFixed a bug in linearmodelElasticNet which caused the input to be overridden when using parametercopyXTrue andcheckinputFalse 10581 by Yacine Mazari
 - FIXFixed a bug in sklearnlinearmodelLasso where the coefficient had wrong shape when fitinterceptFalse 10687 by Martin Hahn
 - FIXFixed a bug in sklearnlinearmodelLogisticRegression where the multiclassmultinomial with binary output with warmstartTrue 10836 by Aishwarya Srinivasan
 - FIXFixed a bug in linearmodelRidgeCV where using integer alphas raised an error 10397 by Mabel VillalbalJiménez
 - FIXFixed condition triggering gap computation in linearmodelLasso andlinearmodelElasticNet when working with sparse matrices 10992 by Alexandre Gramfort
 - FIX Fixed a bug in linearmodelSGDClassifier linearmodelSGDRegressor linearmodelPassiveAggressiveClassifier linearmodelPassiveAggressiveRegressor andlinearmodelPerceptron where the stopping criterion was stopping the algorithm before convergence A parameter niternochange was added and set by default to 5 Previous behavior is equivalent to setting the parameter to 1 9043 by Tom Dupre la Tour
 - FIXFixed a bug where liblinear and libsvmbased estimators would segfault if passed a scipysparse matrix with 64bit indices They now raise a ValueError 11327 by Karan Dhingra and Joel Nothman
 - API (Application Programming Interface) C HANGE The default values of the solver andmulticlass parameters of linearmodelLogisticRegression will change respectively from liblinear andovr in version 020 to lbfgs andauto in version 022 A FutureWarning is raised when the default values are used 11905 by Tom Dupre la Tour and Joel Nothman
 - API (Application Programming Interface) C HANGE DeprecatepositiveTrue option inlinearmodelLars as the underlying implementation is broken Use linearmodelLasso instead 9837 by Alexandre Gramfort
- 54 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- API (Application Programming Interface) C HANGE niter may vary from previous releases in linearmodelLogisticRegression withsolverbfgs andlinearmodelHuberRegressor For Scipy 100 the optimizer could perform more than the requested maximum number of iterations Now both estimators will report at most maxiter iterations even if more were performed 10723 by Joel Nothman

sklearnmanifold

- EFFICIENCY Speed improvements for both ‘exact’ and ‘barneshut’ methods in manifoldTSNE 10593 and 10610 by Tom Dupre la Tour

- FEATURE Support sparse input in manifoldIsomapfit 8554 by Leland McInnes

- FEATURE manifoldtsnetrustworthiness accepts metrics other than Euclidean 9775 by William de Vazelhes

- FIXFixed a bug in manifoldspectralembedding where the normalization of the spectrum was using a division instead of a multiplication 8129 by Jan Margeta Guillaume Lemaitre and Devansh D

- API (Application Programming Interface) C HANGE F EATURE Deprecate precomputed parameter in function manifoldtsne trustworthiness Instead the new parameter metric should be used with any compatible metric including ‘precomputed’ in which case the input matrix Xshould be a matrix of pairwise distances or squared distances 9775 by William de Vazelhes

- API (Application Programming Interface) C HANGE Deprecateprecomputed parameter in function manifoldtsnetrustworthiness Instead the new parameter metric should be used with any compatible metric including ‘precomputed’ in which case the input matrix Xshould be a matrix of pairwise distances or squared distances 9775 by William de Vazelhes

sklearnmetrics

- MAJOR FEATURE Added themetricsdaviesbouldinscore metric for evaluation of clustering models without a ground truth 10827 by Luis Osa

- MAJOR FEATURE Added the metricsbalancedaccuracy score metric and a corresponding balancedaccuracy scorer for binary and multiclass classification 8066 by xyguo and Aman Dalmia and 10587 by Joel Nothman

- FEATURE Partial AUC is available via maxfpr parameter in metricsrocaucscore 3840 by Alexander Niederbühl

- FEATURE A scorer based on metricsbrierscoreloss is also available 9521 by Hanmin Qin

- FEATURE Added control over the normalization in metricsnormalizedmutualinfoscore and metricsadjustedmutualinfoscore via theaveragemethod parameter In version 022 the default normalizer for each will become the arithmetic mean of the entropies of each clustering 11124 by Arya McCarthy

- FEATURE Addedoutputdict parameter in metricsclassificationreport to return classification statistics as dictionary 11160 by Dan Barkhorn

- FEATURE metricsclassificationreport now reports all applicable averages on the given data including micro macro and weighted average as well as samples average for multilabel data 11679 by Alexander Pacha

- FEATURE metricsaverageprecisionscore now supports binary ytrue other than0 1 or 1 1 throughposlabel parameter 9980 by Hanmin Qin

- FEATURE metricslabelrankingaverageprecisionscore now supports sampleweight 10845 by Jose PerezParras Toledano

116 Version 0200 55

scikitlearn user guide Release 0213

- FEATURE Adddenseoutput parameter to metricspairwiselinearkernel When False and both inputs are sparse will return a sparse matrix 10999 by Taylor G Smith
 - EFFICIENCY metricssilhouettescore andmetricssilhouettesamples are more memory efficient and run faster This avoids some reported freezes and MemoryErrors 11135 by Joel Nothman
 - FIXFixed a bug in metricsprecisionrecallfscoresupport when truncated rangenlabels is passed as value for labels 10377 by Gaurav Dhingra
 - FIXFixed a bug due to floating point error in metricsrocaucscore with noninteger sample weights 9786 by Hanmin Qin
 - FIXFixed a bug where metricsroccurve sometimes starts on yaxis instead of 0 0 which is inconsistent with the document and other implementations Note that this will not influence the result from metricsrocaucscore 10093 by alexryndin and Hanmin Qin
 - FIXFixed a bug to avoid integer overflow Casted product to 64 bits integer in metrics mutualinfoscore 9772 by Kumar Ashutosh
 - FIXFixed a bug where metricsaverageprecisionscore will sometimes return nan when sampleweight contains 0 9980 by Hanmin Qin
 - FIXFixed a bug in metricsfowlkesmallowsscore to avoid integer overflow Casted return value ofContingency Matrix toint64 and computed product of square roots rather than square root of product 9515 by Alan Liddell and Manh Dao
 - API (Application Programming Interface) C HANGE Deprecatereorder parameter in metricsauc as it's no longer required for metrics rocaucscore Moreover using reorderTrue can hide bugs due to floating point error in the input 9851 by Hanmin Qin
 - API (Application Programming Interface) C HANGE Inmetricsnormalizedmutualinfoscore andmetrics adjustedmutualinfoscore warn that averagemethod will have a new default value In version 022 the default normalizer for each will become the arithmetic mean of the entropies of each clustering Currently metricsnormalizedmutualinfoscore uses the default of averagemethodgeometric andmetricsadjustedmutualinfoscore uses the default ofaveragemethodmax to match their behaviors in version 019 11124 by Arya McCarthy
 - API (Application Programming Interface) C HANGE Thebatchsize parameter to metricspairwisedistancesargminmin and metricspairwisedistancesargmin is deprecated to be removed in v022 It no longer has any effect as batch size is determined by global workingmemory config See Limiting Working Memory 10280 by Joel Nothman and Aman Dalmia
 - sklearnmixture
 - FEATURE Added function fitpredict toixtureGaussianMixture andmixture GaussianMixture which is essentially equivalent to calling fitand predict 10336 by Shu Haoran and Andrew Peng
 - FIXFixed a bug in mixtureBaseMixture where the reported niter was missing an iteration It affectedmixtureGaussianMixture andmixtureBayesianGaussianMixture 10740 by Erich Schubert and Guillaume Lemaitre
 - FIXFixed a bug in mixtureBaseMixture and its subclasses mixtureGaussianMixture and mixtureBayesianGaussianMixture where thelowerbound was not the max lower bound across all initializations when ninit 1 but just the lower bound of the last initialization 10869 by Aurélien Géron
- 56 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

sklearnmodelselection

- FEATURE Addreturnestimator parameter in modelselectioncrossvalidate to return estimators fitted on each split 9686 by Aurélien Bellet
 - FEATURE Newrefittime attribute will be stored in modelselectionGridSearchCV and modelselectionRandomizedSearchCV ifrefit is set toTrue This will allow measuring the complete time it takes to perform hyperparameter optimization and refitting the best model on the whole dataset 11310 by Matthias Feurer
 - FEATURE Exposeerrorscore parameter in modelselectioncrossvalidate modelselectioncrossvalscore modelselectionlearningcurve and modelselectionvalidationcurve to control the behavior triggered when an error occurs in modelselectionfitandscore 11576 by Samuel O Ronsin
 - FEATURE BaseSearchCV now has an experimental private interface to support customized parameter search strategies through its runsearch method See the implementations in modelselection GridSearchCV andmodelselectionRandomizedSearchCV and please provide feedback if you use this Note that we do not assure the stability of this API (Application Programming Interface) beyond version 020 9599 by Joel Nothman
 - ENHANCEMENT Add improved error message in modelselectioncrossvalscore when multiple metrics are passed in scoring keyword 11006 by Ming Li
 - API (Application Programming Interface) C HANGE The default number of crossvalidation folds cvand the default number of splits nsplits in themodelselectionKfold like splitters will change from 3 to 5 in 022 as 3fold has a lot of variance 11557 by Alexandre Boucaud
 - API (Application Programming Interface) C HANGE The default of iid parameter of modelselectionGridSearchCV and modelselectionRandomizedSearchCV will change from True toFalse in version 022 to correspond to the standard definition of crossvalidation and the parameter will be removed in version 024 altogether This parameter is of greatest practical significance where the sizes of different test sets in crossvalidation were very unequal ie in groupbased CV (Cross-Validation) strategies 9085 by Laurent Direr and Andreas Müller
 - API (Application Programming Interface) C HANGE The default value of the errorscore parameter in modelselectionGridSearchCV andmodelselectionRandomizedSearchCV will change to npNaN in version 022 10677 by Kirill Zhdanovich
 - API (Application Programming Interface) C HANGE Changed ValueError exception raised in modelselectionParameterSampler to a UserWarning for case where the class is instantiated with a greater value of niter than the total space of parameters in the parameter grid niter now acts as an upper bound on iterations 10982 by Juliet Lawton
 - API (Application Programming Interface) C HANGE Invalid input for modelselectionParameterGrid now raises TypeError 10928 by Solutus Immensus
- sklearnmultioutput
- MAJOR FEATURE AddedmultioutputRegressorChain for multitarget regression 9257 by Kumar Ashutosh
- sklearnnaivebayes
- MAJOR FEATURE AddednaivebayesComplementNB which implements the Complement Naive Bayes classifier described in Rennie et al 2003 8190 by Michael A Alcorn
 - FEATURE Addvarsmoothing parameter in naivebayesGaussianNB to give a precise control over variances calculation 9681 by Dmitry Mottl
- 116 Version 0200 57

scikitlearn user guide Release 0213

- FIXFixed a bug in naivebayesGaussianNB which incorrectly raised error for prior list which summed to 1 10005 by Gaurav Dhingra
- FIXFixed a bug in naivebayesMultinomialNB which did not accept vector valued pseudocounts alpha 10346 by Tobias Madsen
- sklearnneighbors
- EFFICIENCY neighborsRadiusNeighborsRegressor and neighborsRadiusNeighborsClassifier are now parallelized according to njobs regardless of algorithm 10887 by Joël Billaud
- EFFICIENCY Nearest neighbors query methods are now more memory efficient when algorithmbrute 11136 by Joel Nothman and Aman Dalmia
- FEATURE Addsampleweight parameter to the fit method of neighborsKernelDensity to enable weighting in kernel density estimation 4394 by Samuel O Ronsin
- FEATURE Novelty detection with neighborsLocalOutlierFactor Add a novelty parameter toneighborsLocalOutlierFactor When novelty is set to True neighborsLocalOutlierFactor can then be used for novelty detection ie predict on new unseen data Available prediction methods are predict decisionfunction andscoresamples By default novelty is set toFalse and only the fitpredict method is available By Albert Thomas
- FIXFixed a bug in neighborsNearestNeighbors where fitting a NearestNeighbors model fails when a the distance metric used is a callable and b the input to the NearestNeighbors model is sparse 9579 by Thomas Kober
- FIXFixed a bug so predict inneighborsRadiusNeighborsRegressor can handle empty neighbors set when using non uniform weights Also raises a new warning when no neighbors are found for samples 9655 by Andreas BjerreNielsen
- FIX EFFICIENCY Fixed a bug in KDTree construction that results in faster construction and querying times 11556 by Jake VanderPlas
- FIXFixed a bug in neighborsKDTree andneighborsBallTree where pickled tree objects would change their type to the super class BinaryTree 11774 by Nicolas Hug
- sklearnneuralnetwork
- FEATURE Add niternochange parameter in neuralnetworkBaseMultilayerPerceptron neuralnetworkMLPRegressor andneuralnetworkMLPClassifier to give control over maximum number of epochs to not meet tol improvement 9456 by Nicholas Nadeau
- FIXFixed a bug in neuralnetworkBaseMultilayerPerceptron neuralnetworkMLPRegressor andneuralnetworkMLPClassifier with newniternochange parameter now at 10 from previously hardcoded 2 9456 by Nicholas Nadeau
- FIXFixed a bug in neuralnetworkMLPRegressor where fitting quit unexpectedly early due to local minima or fluctuations 9456 by Nicholas Nadeau
- sklearnpipeline
- FEATURE Thepredict method of pipelinePipeline now passes keyword arguments on to the pipeline's last estimator enabling the use of parameters such as returnstd in a pipeline with caution 9304 by Breno Freitas

scikitlearn user guide Release 0213

•API (Application Programming Interface) C HANGE pipelineFeatureUnion now supports drop as a transformer to drop features 11144 by Thomas Fan

sklearnpreprocessing

•MAJOR FEATURE ExpandedpreprocessingOneHotEncoder to allow to encode categorical string features as a numeric array using a onehot or dummy encoding scheme and added preprocessing OrdinalEncoder to convert to ordinal integers Those two classes now handle encoding of all feature types also handles stringvalued features and derives the categories based on the unique values in the features instead of the maximum value in the features 9151 and 10521 by Vighnesh Birodkar and Joris Van den Bossche

•MAJOR FEATURE AddedpreprocessingKBinsDiscretizer for turning continuous features into categorical or onehot encoded features 7668 9647 10195 10192 11272 11467 and 11505 by Henry Lin Hanmin Qin Tom Dupre la Tour and Giovanni Giuseppe Costa

•MAJOR FEATURE AddedpreprocessingPowerTransformer which implements the YeoJohnson and BoxCox power transformations Power transformations try to find a set of featurewise parametric transformations to approximately map data to a Gaussian distribution centered at zero and with unit variance This is useful as a variancestabilizing transformation in situations where normality and homoscedasticity are desirable 10210 by Eric Chang and Maniteja Nandana and 11520 by Nicolas Hug

•MAJOR FEATURE NaN values are ignored and handled in the following preprocessing methods preprocessingMaxAbsScaler preprocessingMinMaxScaler preprocessing RobustScaler preprocessingStandardScaler preprocessingPowerTransformer preprocessingQuantileTransformer classes and preprocessingmaxabsscale preprocessingminmaxscale preprocessingrobustscale preprocessing scale preprocessingpowertransform preprocessingquantiletransform functions respectively addressed in issues 11011 11005 11308 11206 11306 and 10437 By Lucija Gregov and Guillaume Lemaître

•FEATURE preprocessingPolynomialFeatures now supports sparse input 10452 by Aman Dalmia and Joel Nothman

•FEATURE preprocessingRobustScaler andpreprocessingrobustscale can be fitted using sparse matrices 11308 by Guillaume Lemaître

•FEATURE preprocessingOneHotEncoder now supports the getfeaturenames method to obtain the transformed feature names 10181 by Nirvan Anjirbag and Joris Van den Bossche

•FEATURE A parameter checkinverse was added to preprocessingFunctionTransformer to ensure that func andinversefunc are the inverse of each other 9399 by Guillaume Lemaître

•FEATURE Thetransform method ofsklearnpreprocessingMultiLabelBinarizer now ignores any unknown classes A warning is raised stating the unknown classes classes found which are ignored 10913 by Rodrigo Agundez

•FIXFixed bugs in preprocessingLabelEncoder which would sometimes throw errors when transform orinversetransform was called with empty arrays 10458 by Mayur Kulkarni

•FIXFix ValueError in preprocessingLabelEncoder when using inversetransform on unseen labels 9816 by Charlie Newey

•FIXFix bug inpreprocessingOneHotEncoder which discarded the dtype when returning a sparse matrix output 11042 by Daniel Morales

•FIXFixfit andpartialfit inpreprocessingStandardScaler in the rare case when withmeanFalse andwithstdFalse which was crashing by calling fit more than once and giving inconsistent results for mean whether the input was a sparse or a dense matrix mean will be set to None 116

Version 0200 59

scikitlearn user guide Release 0213

with both sparse and dense inputs nsamplesseen will be also reported for both input types 11235 by Guillaume Lemaitre

•API (Application Programming Interface) C HANGE Deprecate nvalues andcategoricalfeatures parameters and activefeatures featureindices andnvalues attributes of preprocessing OneHotEncoder Thenvalues parameter can be replaced with the new categories parameter and the attributes with the new categories attribute Selecting the categorical features with the categoricalfeatures parameter is now better supported using the compose ColumnTransformer 10521 by Joris Van den Bossche

•API (Application Programming Interface) C HANGE DeprecatepreprocessingImputer and move the corresponding module to impute SimpleImputer 9726 by Kumar Ashutosh

•API (Application Programming Interface) C HANGE Theaxis parameter that was in preprocessingImputer is no longer present in imputeSimpleImputer The behavior is equivalent to axis0 impute along columns Row wise imputation can be performed with FunctionTransformer eg FunctionTransformerlambda X

SimpleImputerfitttransformXTT 10829 by Guillaume Lemaitre and Gilberto Olimpio

•API (Application Programming Interface) C HANGE The NaN marker for the missing values has been changed between the preprocessing Imputer and the imputeSimpleImputer missingvaluesNaN should now be missingvaluesnpnan 11211 by Jeremie du Boisberranger

•API (Application Programming Interface) C HANGE InpreprocessingFunctionTransformer the default of validate will be from True toFalse in 022 10655 by Guillaume Lemaitre sklearnsvm

•FIXFixed a bug in svmSVC where when the argument kernel is unicode in Python2 the predict_proba method was raising an unexpected TypeError given dense inputs 10412 by Jiongyan Zhang

•API (Application Programming Interface) C HANGE Deprecaterandomstate parameter in svmOneClassSVM as the underlying implementation is not random 9497 by Albert Thomas

•API (Application Programming Interface) C HANGE The default value of gamma parameter of svmSVC NuSVC SVRNuSVR OneClassSVM will change from auto toscale in version 022 to account better for unscaled features 8361 by Gaurav Dhingra and Ting Neo

sklearnntree

•ENHANCEMENT Although private and hence not assured API (Application Programming Interface) stability treecriterion ClassificationCriterion andtreecriterionRegressionCriterion may now be imported and extended 10325 by Camil Staps

•FIXFixed a bug in treeBaseDecisionTree withsplitterbest where split threshold could become infinite when values in X were near infinite 10536 by Jonathan Ohayon

•FIXFixed a bug in treeMAE to ensure sample weights are being used during the calculation of tree MAE impurity Previous behaviour could cause suboptimal splits to be chosen since the impurity calculation considered all samples to be of equal weight importance 11464 by John Stott sklearnutils

•FEATURE utilscheckarray andutilscheckXy now haveacceptlargesparsed to control whether scipysparse matrices with 64bit indices should be rejected 11327 by Karan Dhingra and Joel Nothman

60 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- EFFICIENCY F IXAvoid copying the data in utilscheckarray when the input data is a memmap and copyFalse 10663 by Arthur Mensch and Loïc Estève
- API (Application Programming Interface) C HANGE utilscheckarray yield aFutureWarning indicating that arrays of bytesstrings will be interpreted as decimal numbers beginning in version 022 10229 by Ryan Lee
- Multiple modules
- FEATURE API (Application Programming Interface) C HANGE More consistent outlier detection API (Application Programming Interface) Add a score insvmOneClassSVM ensembleIsolationForest neighborsLocalOutlierFactor
- covarianceEllipticEnvelope It allows to access raw score functions from original parameters A new offset parameter allows to link scoresamples anddecisionfunction methods The contamination parameter of ensembleIsolationForest andneighbors LocalOutlierFactor decisionfunction methods is used to define this offset such that outliers resp inliers have negative resp positive decisionfunction values By default contamination is kept unchanged to 0.1 for a deprecation period In 0.22 it will be set to “auto” thus using methodspecific score offsets In covarianceEllipticEnvelope decisionfunction method the rawvalues parameter is deprecated as the shifted Mahalanobis distance will be always returned in 0.22 9015 by Nicolas Goix
- FEATURE API (Application Programming Interface) C HANGE Abehaviour parameter has been introduced in ensemble IsolationForest to ensure backward compatibility In the old behaviour the decisionfunction is independent of the contamination parameter A threshold attribute depending on the contamination parameter is thus used In the new behaviour the decisionfunction is dependent on the contamination parameter in such a way that 0 becomes its natural threshold to detect outliers Setting behaviour to “old” is deprecated and will not be possible in version 0.22 Beside the behaviour parameter will be removed in 0.24 11553 by Nicolas Goix
- API (Application Programming Interface) C HANGE Added convergence warning to svmLinearSVC andlinearmodel LogisticRegression whenverbose is set to 0 10881 by Alexandre Sevin
- API (Application Programming Interface) C HANGE Changed warning type from UserWarning toexceptionsConvergenceWarning for failing convergence in linearmodellogisticregressionpath linearmodel RANSACRegressor linearmodelridgeregression gaussianprocess GaussianProcessRegressor gaussianprocessGaussianProcessClassifier decompositionfastica crossdecompositionPLSCanonical cluster AffinityPropagation andclusterBirch 10306 by Jonathan Siebert
- Miscellaneous
- MAJOR FEATURE A new configuration parameter workingmemory was added to control memory consumption limits in chunked operations such as the new metricspairwisedistanceschunked See Limiting Working Memory 10280 by Joel Nothman and Aman Dalmia
- FEATURE The version of joblib bundled with Scikitlearn is now 0.12 This uses a new default multiprocessing implementation named loky While this may incur some memory and communication overhead it should provide greater crossplatform stability than relying on Python standard library multiprocessing 11741 by the joblib developers especially Thomas Moreau and Olivier Grisel
- FEATURE An environment variable to use the site joblib instead of the vendored one was added Environment variables The main API (Application Programming Interface) of joblib is now exposed in sklearnutils 11166 by Gael Varoquaux
- FEATURE Add almost complete PyPy 3 support Known unsupported functionalities are datasets loadsvmlightfile featureextractionFeatureHasher andfeatureextraction textHashingVectorizer For running on PyPy PyPy3v510 Numpy 1.14.0 and scipy 1.10.0 are required 11010 by Ronan Lamy and Roman Yurchak
- 116 Version 0.20.0 61

scikitlearn user guide Release 0213

- FEATURE A utility method sklearnshowversions was added to print out information relevant for debugging It includes the user system the Python executable the version of the main libraries and BLAS binding information 11596 by Alexandre Boucaud
- FIXFixed a bug when setting parameters on metaestimator involving both a wrapped estimator and its parameter 9999 by Marcus Voss and Joel Nothman
- FIXFixed a bug where calling sklearnbaseclone was not thread safe and could result in a “pop from empty list” error 9569 by Andreas Müller
- API (Application Programming Interface) CHANGE The default value of njobs is changed from 1toNone in all related functions and classes njobsNone meansunset It will generally be interpreted as njobs1 unless the current joblib Parallel backend context specifies otherwise See Glossary for additional information Note that this change happens immediately ie without a deprecation cycle 11741 by Olivier Grisel
- FIXFixed a bug in validation helpers where passing a Dask DataFrame results in an error 12462 by Zachariah Miller

1165 Changes to estimator checks

These changes mostly affect library developers

- Checks for transformers now apply if the estimator implements transform regardless of whether it inherits from sklearnbaseTransformerMixin 10474 by Joel Nothman
- Classifiers are now checked for consistency between decisionfunction and categorical predictions 10500 by Narine Kokhlikyan
- Allow tests in utilsestimatorcheckscheckestimator to test functions that accept pairwise data 9701 by Kyle Johnson
- Allowutilsestimatorcheckscheckestimator to check that there is no private settings apart from parameters during estimator initialization 9378 by Herilalaina Rakotoarison
- The set of checks in utilsestimatorcheckscheckestimator now includes a checksetparams test which checks that setparams is equivalent to passing parameters in init and warns if it encounters parameter validation 7738 by Alvin Chiang
- Add invariance tests for clustering metrics 8102 by Ankita Sinha and Guillaume Lemaitre
- Addcheckmethodssubsetinvariance tocheckestimator which checks that estimator methods are invariant if applied to a data subset 10428 by Jonathan Ohayon
- Add tests in utilsestimatorcheckscheckestimator to check that an estimator can handle readonly memmap input data 10663 by Arthur Mensch and Loïc Estève
- checksampleweightspandasseries now uses 8 rather than 6 samples to accommodate for the default number of clusters in clusterKMeans 10933 by Johannes Hansen
- Estimators are now checked for whether sampleweightNone equates to sampleweightnp.ones 11558 by Sergul Aydore

1166 Code and Documentation Contributors

Thanks to everyone who has contributed to the maintenance and improvement of the project since version 019 including

211217613 Aarshay Jain absolutelyNoWarranty Adam Greenhall Adam Kleczewski Adam RichieHalford adelr AdityaDaflapurkar Adrin Jalali Aidan Fitzgerald aishgrt1 Akash Shivram Alan Liddell Alan Yee Albert Thomas 62 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

Alexander Lenail AlexanderN Alexandre Boucaud Alexandre Gramfort Alexandre Sevin Alex Egg Alvaro Perez Diaz Amanda Aman Dalmia Andreas BjerreNielsen Andreas Mueller Andrew Peng Angus Williams Aniruddha Dave annaayzshntat Anthony Gitter Antonio Quinonez Anubhav Marwaha Arik Pamnani Arthur Ozga Artiem K Arunava Arya McCarthy Attractadore Aurélien Bellet Aurélien Geron Ayush Gupta Balakumaran Manoharan Bangda Sun Barry Hart Bastian Venthur Ben Lawson Benn Roth Breno Freitas Brent Yi brett koonce Caio Oliveira Camil Staps cclauss Chady Kamar Charlie Brummitt Charlie Newey chris Chris Chris Catalfo Chris Foster Chris Holdgraf Christian Braune Christian Hirsch Christian Hogan Christopher Jenness Clement Joudet cnx cwitte Dallas Card Dan Barkhorn Daniel Daniel Ferreira Daniel Gomez Daniel Klevebring Danielle Shwed Daniel Mohns Danil Baibak Darius Morawiec David Beach David Burns David Kirkby David Nichol son David Pickup Derek Didi BarZev diegodlh Dillon Gardner Dillon Niederhut dilutedsaucedlovell Dmitry Mottl Dmitry Petrov Dor Cohen Douglas Duhaime Ekaterina Tuzova Eric Chang Eric Dean Sanchez Erich Schubert Eunji FangChieh Chou FarahSaeed felix Félix Raimundo fenx filipj8 FrankHui Franz Wompner Freija Descamps frsi Gabriele Calvo Gael Varoquaux Gaurav Dhingra Georgi Peev Gil Forsyth Giovanni Giuseppe Costa gkevinyen5418 goncalorodrigues Gryllos Prokopis Guillaume Lemaitre Guillaume “Vermeille” Sanchez Gustavo De Mari Pereira hakaa1 Hanmin Qin Henry Lin Hong Honghe Hossein Pourbozorg Hristo Hunan Ros tomyan iampat Ivan PANICO Jaewon Chung Jake VanderPlas jakirkham James Bourbeau James Malcolm Jamie Cox Jan Koch Jan Margeta Jan Schlüter janvanrijn Jason Wolosonovich JC Liu Jeb Bearer jeremiedbb Jimmy Wan Jinkun Wang Jiongyan Zhang jjabl jkleint Joan Massich Joël Billaud Joel Nothman Johannes Hansen JohnStott Jonatan Samoocha Jonathan Ohayon Jörg Döpfert Joris Van den Bossche Jose PerezParras Toledano josephsalmon jotasi jschendel Julian Kuhlmann Julien Chaumond julietcl Justin Shenk Karl F Kasper Primdal Lauritzen Katrin Leinweber Kirill ksemb Kuai Yu Kumar Ashutosh Kyeongpil Kang Kye Taylor kyledrogo Leland McInnes Léo DS Liam Geron Liutong Zhou Lizao Li lkjcalc Loic Esteve louib Luciano Viola Lucija Gregov Luis Osa Luis Pedro Coelho Luke M Craig Luke Persola Mabel Mabel Villalba Maniteja Nandana Markl wanchyshyn Mark Roth Markus Müller MarsGuy Martin Gubri martinhahn martinkokos mathurinm Matthias Feuerer Max Copeland Mayur Kulkarni Meghann Agarwal Melanie Goetz Michael A Alcorn Minghui Liu Ming Li Minh Le Mohamed Ali Jamaoui Mohamed Maskani Mohammad Shahebaz Muayyad Alsadi Nabarun Pal Na garjuna Kumar Naoya Kanai Narendran Santhanam NarineK Nathaniel Saul Nathan Suh Nicholas Nadeau PEng A VS Nick Hoh Nicolas Goix Nicolas Hug Nicolau Werneck nielsenmarkus11 Nihar Sheth Nikita Titov Nilesh Kevlani Nirvan Anjirbag notmatthancock nzw Oleksandr Pavlyk oliblum90 Oliver Rausch Olivier Grisel Oren Milman Osaid Rehman Nasir pasbi Patrick Fernandes Patrick Olden Paul Paczuski Pedro Morales Peter Peter St John pierreablin pietruh Pinaki Nath Chowdhury Piotr Szyma ński Pradeep Reddy Raamana Pravara D Mahajan pravarmahajan QingYing Chen Raghav RV Rajendra arora RAKOTOARISON Herilalaina Rameshwar Bhaskaran RankyLau Rasul Kerimov Reiichiro Nakano Rob Roman Kosobrodov Roman Yurchak Ronan Lamy rragundez Rüdiger Busche Ryan Sachin Kelkar Sagnik Bhattacharya Sailesh Choyal Sam Radhakrishnan Sam Steingold Samuel Bell Samuel O Ronsin Saqib Nizam Shamsi SATISH J Saurabh Gupta Scott Gigante Sebastian Flen nerhag Sebastian Raschka Sebastian Dubois Sébastien Lerique Sebastin Santy Sergey Feldman Sergey Melderis Sergul Aydore Shahebaz Shalil Awaleyy Shangwu Yao Sharad Vijalapuram Sharan Yalburgi shenhanc78 Shivam Rastogi Shu Haoran siftikha Sinclert Pérez SolutusImmensus Somya Anand srajan paliwal Sriharsha Hatwar Sri Krishna Stefan van der Walt Stephen McDowell Steven Brown syonekura Taehoon Lee Takanori Hayashi tarcusx Taylor G Smith theriley106 Thomas Thomas Fan Thomas Heavey Tobias Madsen tobycheese Tom Augspurger Tom Dupré la Tour Tommy Trevor Stephens Trishnendu Ghorai Tulio Casagrande twosigmajab Umar Farouk Umar Urvang Patel Utkarsh Upadhyay Vadim Markovtsev Varun Agrawal Vathsala Achar Vilhelm von Ehren heim Vinayak Mehta Vinit Vinod Kumar L Viraj Mavani Viraj Navkal Vivek Kumar Vlad Niculae vqean3 Vris hank Bhardwaj vufg wallygauze Warut Vijitbenjaronk wdevazelhes Wenhao Zhang Wes Barnett Will William de Vazelhes Will Rosenfeld Xin Xiong Yiming Paul Li ymazari Yufeng Zach Griffith Zé Vinícius Zhenqing Hu Zhiqing Xiao Zijie ZJ Poh

117 Previous Releases

1171 Version 0192

July 2018

117 Previous Releases 63

scikitlearn user guide Release 0213

This release is exclusively in order to support Python 37

Related changes

- niter may vary from previous releases in linearmodelLogisticRegression with solverlbfgs andlinearmodelHuberRegressor For Scipy 100 the optimizer could perform more than the requested maximum number of iterations Now both estimators will report at most maxiter iterations even if more were performed 10723 by Joel Nothman

1172 Version 0191

October 23 2017

This is a bugfix release with some minor documentation improvements and enhancements to features released in

0190

Note there may be minor differences in TSNE output in this release due to 9623 in the case where multiple samples have equal distance to some sample

Changelog

API (Application Programming Interface) changes

- Reverted the addition of metricsndcg_score and metrics_sdcg_score which had been merged into version 0190 by error The implementations were broken and undocumented
- return_train_score which was added to model_selection.GridSearchCV model_selection.RandomizedSearchCV and model_selection.cross_validate in version 0190 will be changing its default value from True to False in version 021 We found that calculating training score could have a great effect on cross validation runtime in some cases Users should explicitly set return_train_score to False if prediction or scoring functions are slow resulting in a deleterious effect on CV (Cross-Validation) runtime or to True if they wish to use the calculated scores 9677 by Kumar Ashutosh and Joel Nothman
- correlation_models and regression_models from the legacy gaussian processes implementation have been belatedly deprecated 9717 by Kumar Ashutosh

Bug fixes

- Avoid integer overflows in metrics_matthews_corrcoef 9693 by Sam Steingold
- Fixed a bug in the objective function for manifold_TSNE both exact and with the BarnesHut approximation when n_components > 3 9711 by goncalorodrigues
- Fix regression in model_selection.cross_val_predict where it raised an error with method_predict_proba for some probabilistic classifiers 9641 by James Bourbeau
- Fixed a bug where datasets_make_classification modified its input weights 9865 by Sachin Kelkar
- model_selection.StratifiedShuffleSplit now works with multioutput multiclass or multilabel data with more than 1000 columns 9922 by Charlie Brummitt
- Fixed a bug with nested and conditional parameter setting eg setting a pipeline step and its parameter at the same time 9945 by Andreas Müller and Joel Nothman

64 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

Regressions in 0190 fixed in 0191

- Fixed a bug where parallelised prediction in random forests was not threadsafe and could rarely result in arbitrary errors 9830 by Joel Nothman
- Fix regression in modelselectioncrossvalpredict where it no longer accepted X as a list 9600 by Rasul Kerimov
- Fixed handling of crossvalpredict for binary classification with methoddecisionfunction 9593 by Reiichiro Nakano and core devs
- Fix regression in pipelinePipeline where it no longer accepted steps as a tuple 9604 by Joris Van den Bossche
- Fix bug where niter was not properly deprecated leaving niter unavailable for interim use inlinearmodelSGDClassifier linearmodelSGDRegressor linearmodelPassiveAggressiveClassifier linearmodelPassiveAggressiveRegressor and linearmodelPerceptron 9558 by Andreas Müller
- Dataset fetchers make sure temporary files are closed before removing them which caused errors on Windows 9847 by Joan Massich
- Fixed a regression in manifoldTSNE where it no longer supported metrics other than ‘euclidean’ and ‘pre computed’ 9623 by Oli Blum

Enhancements

- Our test suite and utilsestimatorcheckscheckestimators can now be run without Nose in stalled 9697 by Joan Massich
- To improve usability of version 019’s pipelinePipeline cachingmemory now allows joblib Memory instances This make use of the new utilsvalidationcheckmemory helper issue 9584 by Kumar Ashutosh
- Some fixes to examples 9750 9788 9815
- Made a FutureWarning in SGDbased estimators less verbose 9802 by Vrishank Bhardwaj

Code and Documentation Contributors

With thanks to

Joel Nothman Loic Esteve Andreas Mueller Kumar Ashutosh Vrishank Bhardwaj Hanmin Qin Rasul Kerimov James Bourbeau Nagarjuna Kumar Nathaniel Saul Olivier Grisel Roman Yurchak Reiichiro Nakano Sachin Kelkar Sam Steingold Yaroslav Halchenko diegodlh felix goncalorodrigues jkleint oliblum90 pasbi Anthony Gitter Ben Lawson Charlie Brummitt Didi BarZev Gael Varoquaux Joan Massich Joris Van den Bossche nielsenmarkus11173 Version 019

August 12 2017

Highlights

We are excited to release a number of great new features including neighborsLocalOutlierFactor for anomaly detection preprocessingQuantileTransformer for robust feature transformation and themultioutputClassifierChain metaestimator to simply account for dependencies between classes

117 Previous Releases 65

scikitlearn user guide Release 0213

in multilabel problems We have some new algorithms in existing estimators such as multiplicative up
date indecompositionNMF and multinomial linearmodelLogisticRegression with L1 loss use
solversaga

Cross validation is now able to return the results from multiple metric evaluations The new modelselection
crossvalidate can return many scores on the test data as well as training set performance and timings and we
have extended the scoring andrefit parameters for gridrandomized search to handle multiple metrics
You can also learn faster For instance the new option to cache transformations inpipelinePipeline makes
grid search over pipelines including slow transformations much more efficient And you can predict faster if you're
sure you know what you're doing you can turn off validating that the input is finite using configcontext
We've made some important fixes too We've fixed a longstanding implementation error in metrics
averageprecisionscore so please be cautious with prior results reported from that function A number
of errors in the manifoldTSNE implementation have been fixed particularly in the default BarnesHut approx
imationsemisupervisedLabelSpreading andsemisupervisedLabelPropagation have had
substantial fixes LabelPropagation was previously broken LabelSpreading should now correctly respect its alpha
parameter

Changed models

The following estimators and functions when fit with the same data and parameters may produce different models
from the previous version This often occurs due to changes in the modelling logic bug fixes or enhancements or in
random sampling procedures

- clusterKMeans with sparse X and initial centroids given bug fix
- crossdecompositionPLSRegression withscaleTrue bug fix
- ensembleGradientBoostingClassifier andensembleGradientBoostingRegressor
whereminimpuritysplit is used bug fix
- gradient boosting lossquantile bug fix
- ensembleIsolationForest bug fix
- featureselectionSelectFdr bug fix
- linearmodelRANSACRegressor bug fix
- linearmodelLassoLars bug fix
- linearmodelLassoLarsIC bug fix
- manifoldTSNE bug fix
- neighborsNearestCentroid bug fix
- semisupervisedLabelSpreading bug fix
- semisupervisedLabelPropagation bug fix
- tree based models where minweightfractionleaf is used enhancement
- modelselectionStratifiedKFold withshuffleTrue this change due to 7823 was not men

tioned in the release notes at the time
Details are listed in the changelog below

While we are trying to better inform users by providing this information we cannot assure that this list is complete
66 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

Changelog

New features

Classifiers and regressors

- AddedmultioutputClassifierChain for multilabel classification By Adam Kleczewski
- Added solver saga that implements the improved version of Stochastic Average Gradient in linearmodelLogisticRegression andlinearmodelRidge It allows the use of L1 penalty with multinomial logistic loss and behaves marginally better than ‘sag’ during the first epochs of ridge and logistic regression 8446 by Arthur Mensch

Other estimators

- Added the neighborsLocalOutlierFactor class for anomaly detection based on nearest neighbors 5279 by Nicolas Goix and Alexandre Gramfort
- Added preprocessingQuantileTransformer class and preprocessing quantiletransform function for features normalization based on quantiles 8363 by Denis Engemann Guillaume Lemaître Olivier Grisel Raghav RV Thierry Guillemot and Gael Varoquaux
- The new solver mu implements a Multiply Update in decompositionNMF allowing the optimization of all betadivergences including the Frobenius norm the generalized KullbackLeibler divergence and the ItakuraSaito divergence 5295 by Tom Dupré la Tour

Model selection and evaluation

- modelselectionGridSearchCV andmodelselectionRandomizedSearchCV now support simultaneous evaluation of multiple metrics Refer to the Specifying multiple metrics for evaluation section of the user guide for more information 7388 by Raghav RV
- Added the modelselectioncrossvalidate which allows evaluation of multiple metrics This function returns a dict with more useful information from crossvalidation such as the train scores fit times and score times Refer to The crossvalidate function and multiple metric evaluation section of the userguide for more information 7388 by Raghav RV
- Addedmetricsmeansquaredlogerror which computes the mean square error of the logarithmic transformation of targets particularly useful for targets with an exponential trend 7655 by Karan Desai
- Addedmetricsdcgscore andmetricsndcgsgscore which compute Discounted cumulative gain DCG and Normalized discounted cumulative gain NDCG 7739 by David Gasquez
- Added the modelselectionRepeatedKFold andmodelselection RepeatedStratifiedKFold 8120 by Neeraj Gangwar

Miscellaneous

- Validation that input data contains no NaN or inf can now be suppressed using configcontext at your own risk This will save on runtime and may be particularly useful for prediction time 7548 by Joel Nothman
- Added a test to ensure parameter listing in docstrings match the functionclass signature 9206 by Alexandre Gramfort and Raghav RV

Enhancements

Trees and ensembles

- The minweightfractionleaf constraint in tree construction is now more efficient taking a fast path to declare a node a leaf if its weight is less than 2 the minimum Note that the constructed tree will be different from previous versions where minweightfractionleaf is used 7441 by Nelson Liu

scikitlearn user guide Release 0213

- ensembleGradientBoostingClassifier andensembleGradientBoostingRegressor now support sparse input for prediction 6101 by Ibraim Ganiev
 - ensembleVotingClassifier now allows changing estimators by using ensembleVotingClassifiersetparams An estimator can also be removed by setting it to None 7674 by Yichuan Liu
 - treeexportgraphviz now shows configurable number of decimal places 8698 by Guillaume Lemaitre
 - Addedflattenttransform parameter to ensembleVotingClassifier to change output shape of transform method to 2 dimensional 7794 by Ibraim Ganiev and Herilalaina Rakotoarison
 - Linear kernelized and related models
 - linearmodelSGDClassifier linearmodelSGDRegressor linearmodelPassiveAggressiveClassifier linearmodelPassiveAggressiveRegressor and linearmodelPerceptron now expose maxiter andtol parameters to handle convergence more preciselyniter parameter is deprecated and the fitted estimator exposes a niter attribute with actual number of iterations before convergence 5036 by Tom Dupre la Tour
 - Added average parameter to perform weight averaging in linearmodelPassiveAggressiveClassifier 4939 by Andrea Esuli
 - linearmodelRANSACRegressor no longer throws an error when calling fit if no inliers are found in its first iteration Furthermore causes of skipped iterations are tracked in newly added attributes nskips 7914 by Michael Horrell
 - IngaussianprocessGaussianProcessRegressor method predict is a lot faster with returnstdTrue 8591 by Hadrien Bertrand
 - Addedreturnstd topredict method oflinearmodelARDRegression andlinearmodelBayesianRidge 7838 by Sergey Feldman
 - Memory usage enhancements Prevent cast from float32 to float64 in linearmodelMultiTaskElasticNet linearmodelLogisticRegression when using newtoncg solver and linearmodelRidge when using svd sparsecg cholesky or lsqr solvers 8835 8061 by Joan Massich and Nicolas Cordier and Thierry Guillemot
 - Other predictors
 - Custom metrics for the neighbors binary trees now have fewer constraints they must take two 1darrays and return a float 6288 by Jake Vanderplas
 - algorithmauto inneighbors estimators now chooses the most appropriate algorithm for all input types and metrics 9145 by Herilalaina Rakotoarison and Reddy Chinthala
 - Decomposition manifold learning and clustering
 - clusterMiniBatchKMeans andclusterKMeans now use significantly less memory when assigning data points to their nearest cluster center 7721 by Jon Crall
 - decompositionPCA decompositionIncrementalPCA anddecompositionTruncatedSVD now expose the singular values from the underlying SVD They are stored in the attributesingularvalues like indecompositionIncrementalPCA 7685 by Tommy Löfstedt
 - decompositionNMF now faster when betaloss0 9277 by hongkahjun
 - Memory improvements for method barneshut inmanifoldTSNE 7089 by Thomas Moreau and Olivier Grisel
 - Optimization schedule improvements for BarnesHut manifoldTSNE so the results are closer to the one from the reference implementation lvdmaatenbhtsne by Thomas Moreau and Olivier Grisel
- 68 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Memory usage enhancements Prevent cast from float32 to float64 in decompositionPCA and decompositionrandomizedsvdlowrank 9067 by Raghav RV
 - Preprocessing and feature selection
 - Addednormorder parameter to featureselectionSelectFromModel to enable selection of the norm order when coef is more than 1D 6181 by Antoine Wendlinger
 - Added ability to use sparse matrices in featureselectionfregression withcenterTrue 8065 by Daniel Lejeune
 - Small performance improvement to ngram creation in featureextractiontext by binding methods for loops and specialcasing unigrams 7567 by Jaye Doepeke
 - Relax assumption on the data for the kernelapproximationSkewedChi2Sampler Since the SkewedChi2 kernel is defined on the open interval $-\infty$ to ∞ the transform function should not check whether $X \geq 0$ but whether X selfskewedness 7573 by Romain Brault
 - Made default kernel parameters kerneldependent in kernelapproximationNystroem 5229 by Saurabh Bansod and Andreas Müller
 - Model evaluation and metaestimators
 - pipelinePipeline is now able to cache transformers within a pipeline by using the memory constructor parameter 7990 by Guillaume Lemaitre
 - pipelinePipeline steps can now be accessed as attributes of its namedsteps attribute 8586 by Herilalaina Rakotoarison
 - Addedsampleweight parameter to pipelinePipelinescore 7723 by Mikhail Korobov
 - Added ability to set njobs parameter to pipelinemakeunion ATypeError will be raised for any other kwargs 8028 by Alexander Booth
 - modelselectionGridSearchCV modelselectionRandomizedSearchCV and modelselectioncrossvalscore now allow estimators with callable kernels which were previously prohibited 8005 by Andreas Müller
 - modelselectioncrossvalpredict now returns output of the correct shape for all values of the argumentmethod 7863 by Aman Dalmia
 - Addedshuffle andrandomstate parameters to shuffle training data before taking prefixes of it based on training sizes in modelselectionlearningcurve 7506 by Narine Kokhlikyan
 - modelselectionStratifiedShuffleSplit now works with multioutput multiclass or multilabel data 9044 by Vlad Niculae
 - Speed improvements to modelselectionStratifiedShuffleSplit 5991 by Arthur Mensch and Joel Nothman
 - Addshuffle parameter to modelselectiontraintestsplit 8845 by themrmax
 - multioutputMultiOutputRegressor andmultioutputMultiOutputClassifier now support online learning using partialfit issue8053 by Peng Yu
 - Addmaxtrainsize parameter to modelselectionTimeSeriesSplit 8282 by Aman Dalmia
 - More clustering metrics are now available through metricsgetscorer andscoring parameters 8117 by Raghav RV
 - A scorer based on metricsexplainedvariancescore is also available 9259 by Hanmin Qin
 - Metrics
 - metricsmatthewscorrcoef now support multiclass classification 8094 by Jon Crall
- 117 Previous Releases 69

scikitlearn user guide Release 0213

- Addsampleweight parameter to metricscohenkappascore 8335 by Victor Poughon

Miscellaneous

- utilscheckestimator now attempts to ensure that methods transform predict etc do not set attributes on the estimator 7533 by Ekaterina Krivich
- Added type checking to the acceptspare parameter in utilsvalidation methods This parameter now accepts only boolean string or listtuple of strings acceptspareNone is deprecated and should be replaced by acceptspareFalse 7880 by Josh Karnofsky
- Make it possible to load a chunk of an svmlight formatted file by passing a range of bytes to datasets loadsvmlightfile 935 by Olivier Grisel
- dummyDummyClassifier anddummyDummyRegressor now accept nonfinite features 8931 by Attractadore

Bug fixes

Trees and ensembles

- Fixed a memory leak in trees when using trees with criterionmae 8002 by Raghav RV
- Fixed a bug where ensembleIsolationForest uses an an incorrect formula for the average path length 8549 by Peter Wang
- Fixed a bug where ensembleAdaBoostClassifier throwsZeroDivisionError while fitting data with single class labels 7501 by Dominik Krzeminski
- Fixed a bug in ensembleGradientBoostingClassifier andensemble GradientBoostingRegressor where a float being compared to 00 usingcaused a divide by zero error 7970 by He Chen
- Fix a bug where ensembleGradientBoostingClassifier andensemble GradientBoostingRegressor ignored the minimpuritysplit parameter 8006 by Sebastian Pölsterl
- Fixedoobscore inensembleBaggingClassifier 8936 by Michael Lewis
- Fixed excessive memory usage in prediction for random forests estimators 8672 by Mike Benfield
- Fixed a bug where sampleweight as a list broke random forests in Python 2 8068 by xor
- Fixed a bug where ensembleIsolationForest fails when maxfeatures is less than 1 5732 by Ishank Gulati
- Fix a bug where gradient boosting with lossquantile computed negative errors for negative values of ytrue ypred leading to wrong values when calling call 8087 by Alexis Mignon
- Fix a bug where ensembleVotingClassifier raises an error when a numpy array is passed in for weights 7983 by Vincent Pham
- Fixed a bug where treeexportgraphviz raised an error when the length of featuresnames does not match nfeatures in the decision tree 8512 by Li Li

Linear kernelized and related models

- Fixed a bug where linearmodelRANSACRegressororfit may run until maxiter if it finds a large inlier group early 8251 by aivision2020
- Fixed a bug where naivebayesMultinomialNB andnaivebayesBernoulliNB failed when alpha0 5814 by Yichuan Liu and Herilalaina Rakotoarison

scikitlearn user guide Release 0213

- Fixed a bug where linearmodelLassoLars does not give the same result as the LassoLars implementation available in R lars library 7849 by Jair Montoya Martinez
 - Fixed a bug in linearmodelRandomizedLasso linearmodelLars linearmodelLassoLars linearmodelLarsCV andlinearmodelLassoLarsCV where the parameter precompute was not used consistently across classes and some values proposed in the docstring could raise errors 5359 by Tom Dupre la Tour
 - Fix inconsistent results between linearmodelRidgeCV andlinearmodelRidge when using normalizeTrue 9302 by Alexandre Gramfort
 - Fix a bug where linearmodelLassoLarsfit sometimes left coef as a list rather than an ndarray 8160 by CJ Carey
 - FixlinearmodelBayesianRidgefit to return ridge parameter alpha andlambda consistent with calculated coefficients coef andintercept 8224 by Peter Gedeck
 - Fixed a bug in svmOneClassSVM where it returned floats instead of integer classes 8676 by Vathsala Achar
 - Fix AICBIC criterion computation in linearmodelLassoLarsIC 9022 by Alexandre Gramfort and Mehmet Basbug
 - Fixed a memory leak in our LibLinear implementation 9024 by Sergei Lebedev
 - Fix bug where stratified CV (Cross-Validation) splitters did not work with linearmodelLassoCV 8973 by Paulo Haddad
 - Fixed a bug in gaussianprocessGaussianProcessRegressor when the standard deviation and covariance predicted without fit would fail with a unmeaningful error by default 6573 by Quazi Marufur Rahman and Manoj Kumar
- Other predictors
- FixsemisupervisedBaseLabelPropagation to correctly implement LabelPropagation and LabelSpreading as done in the referenced papers 9239 by Andre Ambrosio Boechat Utkarsh Upadhyay and Joel Nothman
- Decomposition manifold learning and clustering
- Fixed the implementation of manifoldTSNE
 - earlyexageration parameter had no effect and is now used for the first 250 optimization iterations
 - Fixed the AssertionError Tree consistency failed exception reported in 8992
 - Improve the learning schedule to match the one from the reference implementation lvdmaatenbhtsne by Thomas Moreau and Olivier Grisel
 - Fix a bug in decompositionLatentDirichletAllocation where theperplexity method was returning incorrect results because the transform method returns normalized document topic distributions as of version 018 7954 by Gary Foreman
 - Fix output shape and bugs with njobs 1 in decompositionSparseCoder transform and decompositionsparsencode for onedimensional data and one component This also impacts the output shape of decompositionDictionaryLearning 8086 by Andreas Müller
 - Fixed the implementation of explainedvariance indecompositionPCA decomposition RandomizedPCA anddecompositionIncrementalPCA 9105 by Hanmin Qin
 - Fixed the implementation of noisevariance indecompositionPCA 9108 by Hanmin Qin
 - Fixed a bug where clusterDBSCAN gives incorrect result when input is a precomputed sparse matrix with initial rows all zero 8306 by Akshay Gupta
- 117 Previous Releases 71

scikitlearn user guide Release 0213

- Fix a bug regarding fitting clusterKMeans with a sparse array X and initial centroids where X’s means were unnecessarily being subtracted from the centroids 7872 by Josh Karnofsky
- Fixes to the input validation in covarianceEllipticEnvelope 8086 by Andreas Müller
- Fixed a bug in covarianceMinCovDet where inputting data that produced a singular covariance matrix would cause the helper method cstep to throw an exception 3367 by Jeremy Steward
- Fixed a bug in manifoldTSNE affecting convergence of the gradient descent 8768 by David DeTomaso
- Fixed a bug in manifoldTSNE where it stored the incorrect kldivergence 6507 by Sebastian Saeger
- Fixed improper scaling in crossdecompositionPLSRegression withscaleTrue 7819 by jayzed82
- clusterbiclusterSpectralCoclustering and clusterbiclusterSpectralBiclustering fit method conforms with API (Application Programming Interface) by accepting yand returning the object 6126 7814 by Laurent Direr and Maniteja Nandana
- Fix bug where mixturesample methods did not return as many samples as requested 7702 by Levi John Wolf
- Fixed the shrinkage implementation in neighborsNearestCentroid 9219 by Hanmin Qin
- Preprocessing and feature selection
 - For sparse matrices preprocessingnormalize withreturnnormTrue will now raise a NotImplementedError with ‘l1’ or ‘l2’ norm and with norm ‘max’ the norms returned will be the same as for dense matrices 7771 by Ang Lu
 - Fix a bug where featuresselectionSelectFdr did not exactly implement BenjaminiHochberg procedure It formerly may have selected fewer features than it should 7490 by Peng Meng
 - Fixed a bug where linearmodelRandomizedLasso andlinearmodelRandomizedLogisticRegression breaks for sparse input 8259 by Aman Dalmia
 - Fix a bug where featureextractionFeatureHasher mandatorily applied a sparse random projection to the hashed features preventing the use of featureextractiontextHashingVectorizer in a pipeline with featureextractiontextTfidfTransformer 7565 by Roman Yurchak
 - Fix a bug where featuresselectionmutualinfo regression did not correctly use nneighbors 8181 by Guillaume Lemaitre
- Model evaluation and metaestimators
 - Fixed a bug where modelselectionBaseSearchCVinversetransform returnsselfbestestimatortransform instead of selfbestestimatorinversetransform 8344 by Akshay Gupta and Rasmus Eriksson
 - Added classes attribute to modelselectionGridSearchCV modelselectionRandomizedSearchCV gridsearchGridSearchCV and gridsearchRandomizedSearchCV that matches the classes attribute of bestestimator 7661 and 8295 by Alyssa Batula Dylan WernerMeier and Stephen Hoover
 - Fixed a bug where modelselectionvalidationcurve reused the same estimator for each parameter value 7365 by Aleksandr Sandrovskii
 - modelselectionpermutationtestscore now works with Pandas types 5697 by Stijn Tonk
 - Several fixes to input validation in multiclassOutputCodeClassifier 8086 by Andreas Müller
 - multiclassOneVsOneClassifier ‘spartialfit now ensures all classes are provided upfront 6250 by Asish Panda

scikitlearn user guide Release 0213

- FixmultioutputMultiOutputClassifierpredictproba to return a list of 2d arrays rather than a 3d array In the case where different target columns had different numbers of classes a ValueError would be raised on trying to stack matrices with different dimensions 8093 by Peter Bull
 - Cross validation now works with Pandas datatypes that that have a readonly index 9507 by Loic Esteve Metrics
 - metricsaverageprecisionscore no longer linearly interpolates between operating points and in stead weighs precisions by the change in recall since the last operating point as per the Wikipedia entry 7356 By Nick Dingwall and Gael Varoquaux
 - Fix a bug in metricsclassificationchecktargets which would return binary ifytrue andypred were bothbinary but the union of ytrue andypred wasmulticlass 8377 by Loic Esteve
 - Fixed an integer overflow bug in metricsconfusionmatrix and hence metrics cohenkappascore 8354 7929 by Joel Nothman and Jon Crall
 - Fixed passing of gamma parameter to the chi2 kernel inmetricspairwisepairwise kernels 5211 by Nick Rhinehart Saurabh Bansod and Andreas Müller
- Miscellaneous
- Fixed a bug when datasetsmakeclassification fails when generating more than 30 features 8159 by Herilalaina Rakotoarison
 - Fixed a bug where datasetsmakemoons gives an incorrect result when nsamples is odd 8198 by Josh Levy
 - Somefetch functions in datasets were ignoring the downloadifmissing keyword 7944 by Ralf Gommers
 - Fix estimators to accept a sampleweight parameter of type pandasSeries in theirfit function 7825 by Kathleen Chen
 - Fix a bug in cases where numpycumsum may be numerically unstable raising an exception if instability is identified 7376 and 7331 by Joel Nothman and yangarbiter
 - Fix a bug where baseBaseEstimatorgetstate obstructed pickling customizations of child classes when used in a multiple inheritance context 8316 by Holger Peters
 - Update SphinxGallery from 014 to 017 for resolving links in documentation build with Sphinx15 8010 7986 by Oscar Najera
 - Adddatahome parameter to sklearndatasetsfetchkddcup99 9289 by Loic Esteve
 - Fix dataset loaders using Python 3 version of makedirs to also work in Python 2 9284 by Sebastin Santy
 - Several minor issues were fixed with thanks to the alerts of lgtmcomhttpslgtmcom 9278 by Jean Helie among others

API (Application Programming Interface) changes summary

Trees and ensembles

- Gradient boosting base models are no longer estimators By Andreas Müller
- All tree based estimators now accept a minimpuritydecrease parameter in lieu of the minimpuritysplit which is now deprecated The minimpuritydecrease helps stop splitting the nodes in which the weighted impurity decrease from splitting is no longer at least minimpuritydecrease 8449 by Raghav RV 117 Previous Releases 73

scikitlearn user guide Release 0213

Linear kernelized and related models

- niter parameter is deprecated in linearmodelSGDClassifier linearmodelSGDRegressor linearmodelPassiveAggressiveClassifier linearmodelPassiveAggressiveRegressor and linearmodelPerceptron By Tom Dupre la Tour
- Other predictors

• neighborsLSHForest has been deprecated and will be removed in 021 due to poor performance 9078 by Laurent Direr

• neighborsNearestCentroid no longer purports to support metricprecomputed which now raises an error 8515 by Sergul Aydore

• Thealpha parameter of semisupervisedLabelPropagation now has no effect and is deprecated to be removed in 021 9239 by Andre Ambrosio Boechat Utkarsh Upadhyay and Joel Nothman

Decomposition manifold learning and clustering

- Deprecate the doctopicdistr argument of the perplexity method in decompositionLatentDirichletAllocation because the user no longer has access to the unnormalized document topic distribution needed for the perplexity calculation 7954 by Gary Foreman

- Thentopics parameter of decompositionLatentDirichletAllocation has been renamed to ncomponents and will be removed in version 021 8922 by Attractadore

- decompositionSparsePCATransform 'sridgealpha parameter is deprecated in preference for class parameter 8137 by Naoya Kanai

- clusterDBSCAN now has ametricparams parameter 8139 by Naoya Kanai

Preprocessing and feature selection

- featureselectionSelectFromModel now has a partialfit method only if the underlying estimator does By Andreas Müller

- featureselectionSelectFromModel now validates the threshold parameter and sets the threshold attribute during the call to fit and no longer during the call to transform By Andreas Müller

- Thenonnegative parameter in featureextractionFeatureHasher has been deprecated and replaced with a more principled alternative alternatesign 7565 by Roman Yurchak

- linearmodelRandomizedLogisticRegression and linearmodelRandomizedLasso have been deprecated and will be removed in version 021 8995 by RamanaS

Model evaluation and metaestimators

- Deprecate the fitparams constructor input to the modelselectionGridSearchCV and modelselectionRandomizedSearchCV in favor of passing keyword parameters to the fit methods

of those classes Data dependent parameters needed for model training should be passed as keyword arguments to fit and conforming to this convention will allow the hyperparameter selection classes to be used with tools such as modelselectioncrossvalpredict 2879 by Stephen Hoover

- In version 021 the default behavior of splitters that use the testsize and trainsize parameter will change such that specifying trainsize alone will cause testsize to be the remainder 7459 by Nelson Liu

- multiclassOneVsRestClassifier now has partialfit decisionfunction and predict_proba methods only when the underlying estimator does 7812 by Andreas Müller and Mikhail Korobov

- multiclassOneVsRestClassifier now has a partialfit method only if the underlying estimator does By Andreas Müller

scikitlearn user guide Release 0213

- The decision function output shape for binary classification in multiclass OneVsRestClassifier and multiclass OneVsOneClassifier is now n_samples to conform to scikitlearn conventions 9100 by Andreas Müller
 - The multioutput MultiOutputClassifier.predict_proba function used to return a 3d array n_samples n_classes n_outputs. In the case where different target columns had different numbers of classes a ValueError would be raised on trying to stack matrices with different dimensions. This function now returns a list of arrays where the length of the list is n_outputs and each array is n_samples n_classes for that particular output 8093 by Peter Bull
 - Replace attribute named_steps dict to utils.Bunch in pipeline Pipeline to enable tab completion in interactive environment. In the case of conflict value on named_steps and dict attribute dict behavior will be prioritized 8481 by Herilalaina Rakotoarison
- Miscellaneous
- Deprecate the y parameter in transform and inverse_transform. The method should not accept y parameter as it's used at the prediction time 8174 by Tahar Zanouda, Alexandre Gramfort and Raghav RV
 - SciPy 0.13.3 and NumPy 1.8.2 are now the minimum supported versions for scikitlearn. The following backported functions in utils have been removed or deprecated accordingly 8854 and 8874 by Naoya Kanai

- The store_covariances and covariances parameters of discriminant analysis QuadraticDiscriminantAnalysis has been renamed to store_covariance and covariance to be consistent with the corresponding parameter names of the discriminant analysis LinearDiscriminantAnalysis. They will be removed in version 0.21 7998 by Jiacheng

Removed in 0.19

- utils.fixes.argmaxpartition
- utils.fixes.array_equal
- utils.fixes.astype
- utils.fixes.bincount
- utils.fixes.expit
- utils.fixes.frombufferempty
- utils.fixes.in1d
- utils.fixes.norm
- utils.fixes.rankdata
- utils.fixes.safe_copy

Deprecated in 0.19 to be removed in 0.21

- utils.srpackeigs
- utils.srpackeigsh
- utils.srpacksvds
- utils.srpackfastdot
- utils.srpacklogsumexp
- utils.srpacknorm
- utils.srpackpinvh
- utils.graphgraphlaplacian

117 Previous Releases 75

scikitlearn user guide Release 0213

-utilsrandomchoice

-utilssparsetoolsconnectedcomponents

-utilsstatsrankdata

- Estimators with both methods decisionfunction andpredictproba are now required to have a monotonic relation between them The method checkdecisionprobaconsistency has been added inutilessestimatorchecks to check their consistency 7578 by Shubham Bhardwaj

- All checks in utilsestimatorchecks in particular utilsestimatorchecks

checkestimator now accept estimator instances Most other checks do not accept estimator classes any more 9019 by Andreas Müller

- Ensure that estimators’ attributes ending with are not set in the constructor but only in the fit method Most notably ensemble estimators deriving from ensembleBaseEnsemble now only have self estimators available after fit 7464 by Lars Buitinck and Loic Esteve

Code and Documentation Contributors

Thanks to everyone who has contributed to the maintenance and improvement of the project since version 018 including

Joel Nothman Loic Esteve Andreas Mueller Guillaume Lemaitre Olivier Grisel Hanmin Qin Raghav RV Alexandre Gramfort themrmax Aman Dalmia Gael Varoquaux Naoya Kanai Tom Dupré la Tour Rishikesh Nelson Liu Tae hoon Lee Nelle Varoquaux Aashil Mikhail Korobov Sebastin Santy Joan Massich Roman Yurchak RAKOTOARI SON Herilalaina Thierry Guillemot Alexandre Abadie Carol Willing Balakumaran Manoharan Josh Karnofsky Vlad Niculae Utkarsh Upadhyay Dmitry Petrov Minghui Liu Srivatsan Vincent Pham Albert Thomas Jake Van derPlas Attractadore JC Liu alexandercb booth chkoar Óscar Nájera Aarshay Jain Kyle Gilliam Ramana Subra manyam CJ Carey Clement Joudet David Robles He Chen Joris Van den Bossche Karan Desai Katie Luangkote Leland McInnes Maniteja Nandana Michele Lacchia Sergei Lebedev Shubham Bhardwaj akshay0724 omtcyfz rickiepark waterponey Vathsala Achar jbDelafosse Ralf Gommers Ekaterina Krivich Vivek Kumar Ishank Gulati Dave Elliott Idirer Reiichiro Nakano Levi John Wolf Mathieu Blondel Sid Kapur Dougal J Sutherland midinas mikebenfield Sourav Singh Aseem Bansal Ibraim Ganiev Stephen Hoover AishwaryaRK Steven C Howell Gary Foreman Neeraj Gangwar Tahar Jon Crall dokato Kathy Chen ferria Thomas Moreau Charlie Brummitt Nicolas Goix Adam Kleczewski Sam Shleifer Nikita Singh Basil Beirouti Giorgio Patrini Manoj Kumar Rafael Possas James Bourbeau James A Bednar Janine Harper Jaye Jean Helie Jeremy Steward Artsiom John Wei Jonathan Ligo Jonathan Rahn seanpwilliams Arthur Mensch Josh Levy Julian Kuhlmann Julien Aubert Jörn Hees Kai shivamgargsya Kat Hempstalk Kaushik Lakshmikanth Kennedy Kenneth Lyons Kenneth Myers Kevin Yap Kir ill Bobyrev Konstantin Podshumok Arthur Imbert Lee Murray toastedcornflakes Lera Li Li Arthur Douillard Mainak Jas tobycheese Manraj Singh Manvendra Singh Marc Meketon MarcoFalke Matthew Brett Matthias Gilch Mehul Ahuja Melanie Goetz Meng Peng Michael Dezube Michal Baumgartner vibrantabhi19 Artem Golu bin Milen Paskov Antonin Carette Morikko MrMjauh NALEPA Emmanuel Namiya Antoine Wendlinger Narine Kokhlikyan NarineK Nate Guerin Angus Williams Ang Lu Nicole Vavrova Nitish Pandey Okhlopkov Daniil Olegovich Andy Craze Om Prakash Parminder Singh Patrick Carlson Patrick Pei Paul Ganssle Paulo Haddad Paweł Lorek Peng Yu Pete Bachant Peter Bull Peter Csiszek Peter Wang Pieter Arthur de Jong PingYao Chang Preston Parry Puneet Mathur Quentin Hibon Andrew Smith Andrew Jackson Ikastner Rameshwar Bhaskaran Re becca Bilbro Remi Rampin Andrea Esuli Rob Hall Robert Bradshaw Romain Brault Aman Pratik Ruifeng Zheng Russell Smith Sachin Agarwal Sailesh Choyal Samson Tan Samuël Weber Sarah Brown Sebastian Pölsterl Sebastian Raschka Sebastian Saeger Alyssa Batula Abhyuday Pratap Singh Sergey Feldman Sergul Aydore Sharan Yalburgi willduan Siddharth Gupta Sri Krishna Almer Stijn Tonk Allen Riddell Theofilos Papapanagiotou Alison Alexis Mignon Tommy Boucher Tommy Löfstedt Toshihiro Kamishima Tyler Folkman Tyler Lanigan Alexander Junge Varun Shenoy Victor Poughon Vilhelm von Ehrenheim Aleksandr Sandrovskii Alan Yee Vlasios Vasileiou Warut Vijitbenjaronk Yang Zhang Yaroslav Halchenko Yichuan Liu Yuichi Fujikawa affanv14 aivision2020 xor andreh7 brady salz campustrampus Agamemnon Krasoulis ditenberg elenasharova filipj8 fukatani gedeck guin iol guoci hakaa1 hongkahjun iamxhy jakirkham jaroslawweber jayzed82 jeroko jmontoyam jonathanstriebe 76 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

josephsalmon jschendel leereeves martinbahn mathurinm mehaksachdeva mlewis1729 mlliou112 mthorrell
ndingwall nuffe yangarbiter plagree pldtc325 Breno Freitas Brett Olsen Brian A Alfano Brian Burns polmauri
Brandon Carter Charlton Austin Chayant T15h Chinmaya Pancholi Christian Danielsen Chung Yen ChyiKwei
Yau pravarmahajan DOHMATOB Elvis Daniel Lejeune Daniel Hnyk Darius Morawiec David DeTomaso David
Gasquez David Haberthür David Heryanto David Kirkby David Nicholson rashchedrin Deborah Gertrude Digges
Denis Engemann Devansh D Dickson Bob Baxley Don86 E LynchKlarup Ed Rogers Elizabeth Ferriss Ellen
Co2 Fabian Egli FangChieh Chou Bing Tian Dai Greg Stupp Grzegorz Szpak Bertrand Thirion Hadrien Bertrand
Harizo Rajaona zxcvbnus Henry Lin Holger Peters Icyblade Dai Igor Andriushchenko Ilya Isaac Laughlin Iván
Vallés Aurélien Bellet JPFrancoia Jacob Schreiber Asish Mahapatra

1174 Version 0182

June 20 2017

Last release with Python 26 support

Scikitlearn 018 is the last major release of scikitlearn to support Python 26 Later versions of scikitlearn will
require Python 27 or above

Changelog

- Fixes for compatibility with NumPy 1130 7946 8355 by Loic Esteve
- Minor compatibility changes in the examples 9010 8040 9149

Code Contributors

Aman Dalmia Loic Esteve Nate Guerin Sergei Lebedev

1175 Version 0181

November 11 2016

Changelog

Enhancements

- Improved samplewithoutreplacement speed by utilizing numpyrandompermutation for most cases

As a result samples may differ in this release for a fixed random state Affected estimators

-ensembleBaggingClassifier

-ensembleBaggingRegressor

-linearmodelRANSACRegressor

-modelselectionRandomizedSearchCV

-randomprojectionSparseRandomProjection

This also affects the datasetsmakeclassification method

117 Previous Releases 77

scikitlearn user guide Release 0213

Bug fixes

- Fix issue where mingradnorm andniterwithoutprogress parameters were not being utilised bymanifoldTSNE 6497 by Sebastian Säger
 - Fix bug for svm’s decision values when decisionfunctionshape isovr insvmSVC svmSVC ’s decisionfunction was incorrect from versions 0170 through 0180 7724 by Bing Tian Dai
 - Attribute explainedvarianceratio of discriminantanalysis LinearDiscriminantAnalysis calculated with SVD and Eigen solver are now of the same length 7632 by JPFrancoia
 - Fixes issue in Univariate feature selection where score functions were not accepting multilabel targets 7676 by Mohammed Affan
 - Fixed setting parameters when calling fit multiple times on featureselectionSelectFromModel 7756 by Andreas Müller
 - Fixes issue in partialfit method ofmulticlassOneVsRestClassifier when number of classes used inpartialfit was less than the total number of classes in the data 7786 by Srivatsan Ramesh
 - Fixes issue in calibrationCalibratedClassifierCV where the sum of probabilities of each class for a data was not 1 and CalibratedClassifierCV now handles the case where the training set has less number of classes than the total data 7799 by Srivatsan Ramesh
 - Fix a bug where sklearnfeatureselectionSelectFdr did not exactly implement Benjamini Hochberg procedure It formerly may have selected fewer features than it should 7490 by Peng Meng
 - sklearnmanifoldLocallyLinearEmbedding now correctly handles integer inputs 6282 by Jake Vanderplas
 - Them inweightfractionleaf parameter of treebased classifiers and regressors now assumes uniform sample weights by default if the sampleweight argument is not passed to the fit function Previously the parameter was silently ignored 7301 by Nelson Liu
 - Numerical issue with linearmodelRidgeCV on centered data when nfeatures nsamples 6178 by Bertrand Thirion
 - Tree splitting criterion classes’ cloningpickling is now memory safe 7680 by Ibraim Ganiev
 - Fixed a bug where decompositionNMF sets itsnitters attribute in transform 7553 by Ekaterina Krivich
 - sklearnlinearmodelLogisticRegressionCV now correctly handles string labels 5874 by Raghav RV
 - Fixed a bug where sklearnmodelselectiontraintestsplit raised an error when stratify is a list of string labels 7593 by Raghav RV
 - Fixed a bug where sklearnmodelselectionGridSearchCV andsklearn modelselectionRandomizedSearchCV were not pickleable because of a pickling bug in np.maMaskedArray 7594 by Raghav RV
 - All crossvalidation utilities in sklearnmodelselection now permit one time crossvalidation splitters for thecvparameter Also nondeterministic crossvalidation splitters where multiple calls to split produce dissimilar splits can be used as cvparameter The sklearnmodelselectionGridSearchCV will crossvalidate each parameter setting on the split produced by the first split call to the crossvalidation splitter 7660 by Raghav RV
 - Fix bug where preprocessingMultiLabelBinarizerfittransform returned an invalid CSR (Compressed Sparse Row) matrix 7750 by CJ Carey
- 78 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Fixed a bug where metricspairwisecosinedistances could return a small negative distance 7732 by Artsion

API (Application Programming Interface) changes summary

Trees and forests

- The minweightfractionleaf parameter of treebased classifiers and regressors now assumes uniform sample weights by default if the sampleweight argument is not passed to the fit function Previously the parameter was silently ignored 7301 by Nelson Liu
- Tree splitting criterion classes' cloningpickling is now memory safe 7680 by Ibraim Ganiev

Linear kernelized and related models

- Length of explainedvarianceratio ofdiscriminantanalysis LinearDiscriminantAnalysis changed for both Eigen and SVD solvers The attribute has now a length of minncomponents nclasses 1 7632 by JPFrancoia
- Numerical issue with linearmodelRidgeCV on centered data when nfeatures nsamples 6178 by Bertrand Thirion

1176 Version 018

September 28 2016

Last release with Python 26 support

Scikitlearn 018 will be the last version of scikitlearn to support Python 26 Later versions of scikitlearn will require Python 27 or above

Model Selection Enhancements and API (Application Programming Interface) Changes

- The modelselection module

The new module sklearnmodelselection which groups together the functionalities of formerly sklearncrossvalidation sklearngridsearch andsklearnlearningcurve introduces new possibilities such as nested crossvalidation and better manipulation of parameter searches with Pandas

Many things will stay the same but there are some key differences Read below to know more about the changes

- Dataindependent CV (Cross-Validation) splitters enabling nested crossvalidation

The new crossvalidation splitters defined in the sklearnmodelselection are no longer initialized with any datadependent parameters such as y Instead they expose a split method that takes in the data and yields a generator for the different splits

This change makes it possible to use the crossvalidation splitters to perform nested crossvalidation facilitated bymodelselectionGridSearchCV andmodelselectionRandomizedSearchCV utilities

- The enhanced cvresults attribute

The newcvresults attribute of modelselectionGridSearchCV andmodelselection RandomizedSearchCV introduced in lieu of the gridscores attribute is a dict of 1D arrays with elements in each array corresponding to the parameter settings ie search candidates

117 Previous Releases 79

scikitlearn user guide Release 0213

Thecvresults dict can be easily imported into pandas as aDataFrame for exploring the search results

Thecvresults arrays include scores for each crossvalidation split with keys such as

split0testscore as well as their mean meantestscore and standard deviation

stdtestscore

The ranks for the search candidates based on their mean crossvalidation score is available at

cvresultsranktestscore

The parameter values for each parameter is stored separately as numpy masked object arrays The value for

that search candidate is masked if the corresponding parameter is not applicable Additionally a list of all the

parameter dicts are stored at cvresultsparams

•Parameters nfolds and niter renamed to nsplits

Some parameter names have changed The nfolds parameter in new modelselectionKFold

modelselectionGroupKFold see below for the name change and modelselection

StratifiedKFold is now renamed to nsplits Theniter parameter in modelselection

ShuffleSplit the new class modelselectionGroupShuffleSplit andmodelselection

StratifiedShuffleSplit is now renamed to nsplits

•Rename of splitter classes which accepts group labels along with data

The crossvalidation splitters LabelKFold LabelShuffleSplit LeaveOneLabelOut and

LeavePLabelOut have been renamed to modelselectionGroupKFold modelselection

GroupShuffleSplit modelselectionLeaveOneGroupOut andmodelselection

LeavePGroupsOut respectively

Note the change from singular form to plural form in modelselectionLeavePGroupsOut

•Fit parameter labels renamed to groups

Thelabels parameter in the split method of the newly renamed splitters modelselection

GroupKFold modelselectionLeaveOneGroupOut modelselection

LeavePGroupsOut modelselectionGroupShuffleSplit is renamed to groups following the

new nomenclature of their class names

•Parameter nlabels renamed to ngroups

The parameter nlabels in the newly renamed modelselectionLeavePGroupsOut is changed to

ngroups

• Training scores and Timing information

cvresults also includes the training scores for each crossvalidation split with keys such

assplit0trainscore as well as their mean meantrainscore and stan

dard deviation stdtrainscore To avoid the cost of evaluating training score set

returntrainscoreFalse

Additionally the mean and standard deviation of the times taken to split train and score the model across all the

crossvalidation splits is available at the key meantime andstdtime respectively

Changelog

New features

Classifiers and Regressors

• The Gaussian Process module has been reimplemented and now offers classification and regression esti

matoms through gaussianprocessGaussianProcessClassifier andgaussianprocess

GaussianProcessRegressor Among other things the new implementation supports kernel engineering

80 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

gradientbased hyperparameter optimization or sampling of functions from GP prior and GP posterior Extensive documentation and examples are provided By Jan Hendrik Metzen

- Added new supervised learning algorithm Multilayer Perceptron 3204 by Issam H Laradji
- AddedlinearmodelHuberRegressor a linear model robust to outliers 5291 by Manoj Kumar
- Added the multioutputMultiOutputRegressor metaestimator It converts single output regressors to multioutput regressors by fitting one regressor per output By Tim Head

Other estimators

- NewmixtureGaussianMixture andmixtureBayesianGaussianMixture replace former mixture models employing faster inference for sounder results 7295 by Wei Xue and Thierry Guillemot
- ClassdecompositionRandomizedPCA is now factored into decompositionPCA and it is available calling with parameter svdsolverrandomized The default number of niter for randomized has changed to 4 The old behavior of PCA (Principal Component Analysis) is recovered by svdsolverfull An additional solver calls arpack and performs truncated nonrandomized SVD By default the best solver is selected depending on the size of the input and the number of components requested 5299 by Giorgio Patrini
- Added two functions for mutual information estimation featuresselection mutualinfoclassif andfeaturesselectionmutualinfo regression These functions can be used in featuresselectionSelectKBest andfeaturesselectionSelectPercentile as score functions By Andrea Bravi and Nikolay Mayorov
- Added the ensembleIsolationForest class for anomaly detection based on random forests By Nicolas Goix

- Addedalgorithmelkan toclusterKMeans implementing Elkan’s fast KMeans algorithm By Andreas Müller

Model selection and evaluation

- Addedmetricsclusterfowlkesmallowsscore the Fowlkes Mallows Index which measures the similarity of two clusterings of a set of points By Arnaud Fouchet and Thierry Guillemot
- Addedmetricscalinskiharabazscore which computes the Calinski and Harabaz score to evaluate the resulting clustering of a set of points By Arnaud Fouchet and Thierry Guillemot
- Added new crossvalidation splitter modelselectionTimeSeriesSplit to handle time series data 6586 by YenChen Lin
- The crossvalidation iterators are replaced by crossvalidation splitters available from sklearn modelselection allowing for nested crossvalidation See Model Selection Enhancements and API (Application Programming Interface) Changes for more information 4294 by Raghav RV

Enhancements

Trees and ensembles

- Added a new splitting criterion for treeDecisionTreeRegressor the mean absolute error This criterion can also be used in ensembleExtraTreesRegressor ensemble RandomForestRegressor and the gradient boosting estimators 6667 by Nelson Liu
- Added weighted impuritybased early stopping criterion for decision tree growth 6954 by Nelson Liu
- The random forest extra tree and decision tree estimators now has a method decisionpath which returns the decision path of samples in the tree By Arnaud Joly
- A new example has been added unveiling the decision tree structure By Arnaud Joly

scikitlearn user guide Release 0213

- Random forest extra trees decision trees and gradient boosting estimator accept the parameter minsamplesplit and minsamplesleaf provided as a percentage of the training samples By yelite and Arnaud Joly
 - Gradient boosting estimators accept the parameter criterion to specify to splitting criterion used in built decision trees 6667 by Nelson Liu
 - The memory footprint is reduced sometimes greatly for ensemblebaggingBaseBagging and classes that inherit from it ie ensembleBaggingClassifier ensembleBaggingRegressor and ensembleIsolationForest by dynamically generating attribute estimatorssamples only when it is needed By David Staub
 - Addednjobs andsampleweight parameters for ensembleVotingClassifier to fit underlying estimators in parallel 5805 by Ibraim Ganiev
 - Linear kernelized and related models
 - InlinearmodelLogisticRegression the SAG solver is now available in the multinomial case 5251 by Tom Dupre la Tour
 - linearmodelRANSACRegressor svmLinearSVC andsvmLinearSVR now support sampleweight By Imaculate
 - Add parameter loss tolinearmodelRANSACRegressor to measure the error on the samples for every trial By Manoj Kumar
 - Prediction of outofsample events with Isotonic Regression isotonicIsotonicRegression is now much faster over 1000x in tests with synthetic data By Jonathan Arfa
 - Isotonic regression isotonicIsotonicRegression now uses a better algorithm to avoid On2 behavior in pathological cases and is also generally faster 6691 By Antony Lee
 - naivebayesGaussianNB now accepts dataindependent classpriors through the parameter priors By Guillaume Lemaitre
 - linearmodelElasticNet andlinearmodelLasso now works with npfloat32 input data without converting it into npfloat64 This allows to reduce the memory consumption 6913 by YenChen Lin
 - semisupervisedLabelPropagation andsemisupervisedLabelSpreading now accept arbitrary kernel functions in addition to strings knn andrbf 5762 by Utkarsh Upadhyay
 - Decomposition manifold learning and clustering
 - Addedinversetransform function to decompositionNMF to compute data matrix of original shape By Anish Shah
 - clusterKMeans andclusterMiniBatchKMeans now works with npfloat32 andnp float64 input data without converting it This allows to reduce the memory consumption by using np float32 6846 by Sebastian Säger and YenChen Lin
 - Preprocessing and feature selection
 - preprocessingRobustScaler now accepts quantilerange parameter 5929 by Konstantin Pod shumok
 - featureextractionFeatureHasher now accepts string values 6173 by Ryad Zenine and Devashish Deshpande
 - Keyword arguments can now be supplied to func inpreprocessingFunctionTransformer by means of the kwargs parameter By Brian McFee
 - featureselectionSelectKBest andfeatureselectionSelectPercentile now accept score functions that take X y as input and return only the scores By Nikolay Mayorov
- 82 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

Model evaluation and metaestimators

- multiclassOneVsOneClassifier andmulticlassOneVsRestClassifier now support partialfit By Asish Panda and Philipp Dowling
- Added support for substituting or disabling pipelinePipeline andpipelineFeatureUnion components using the setparams interface that powers sklearngridsearch See Selecting dimensionality reduction with Pipeline and GridSearchCV By Joel Nothman and Robert McGibbon
- The new cvresults attribute of modelselectionGridSearchCV andmodelselectionRandomizedSearchCV can be easily imported into pandas as a DataFrame Ref Model Selection Enhancements and API (Application Programming Interface) Changes for more information 6697 by Raghav RV
- Generalization of modelselectioncrossvalpredict One can pass method names such as predictproba to be used in the cross validation framework instead of the default predict By Ori Ziv and Sears Merritt
- The training scores and time taken for training followed by scoring for each search candidate are now available at thecvresults dict See Model Selection Enhancements and API (Application Programming Interface) Changes for more information 7325 by Eugene Chen and Raghav RV

Metrics

- Addedlabels flag tometricslogloss to explicitly provide the labels when the number of classes in ytrue andypred differ 7239 by Hong Guangguo with help from Mads Jensen and Nelson Liu
- Support sparse contingency matrices in cluster evaluation metricsclustersupervised to scale to a large number of clusters 7419 by Gregory Stupp and Joel Nothman
- Addsampleweight parameter to metricsmatthewscorrcoef By Jatin Shah and Raghav RV
- Speed up metricssilhouettescore by using vectorized operations By Manoj Kumar
- Addsampleweight parameter to metricsconfusionmatrix By Bernardo Stein

Miscellaneous

- Addednjobs parameter to featureselectionRFECV to compute the score on the test folds in parallel By Manoj Kumar
- Codebase does not contain CC cython generated files they are generated during build Distribution packages will still contain generated CC files By Arthur Mensch
- Reduce the memory usage for 32bit float input arrays of utilssparsefuncmeanvarianceaxis andutilssparsefuncincrmearianceaxis by supporting cython fused types By YenChen Lin

- Theignorewarnings now accept a category argument to ignore only the warnings of a specified type By Thierry Guillemot

- Added parameter returnXy and return type data target tuple option toloadiris dataset 7049 loadbreastcancer dataset 7152 loaddigits datasetloaddiabetes dataset loadlinnerud datasetloadboston dataset 7154 by Manvendra Singh

- Simplification of the clone function deprecate support for estimators that modify parameters in init 5540 by Andreas Müller
- When unpickling a scikitlearn estimator in a different version than the one the estimator was trained with a UserWarning is raised see the documentation on model persistence for more details 7248 By Andreas Müller

scikitlearn user guide Release 0213

Bug fixes

Trees and ensembles

- Random forest extra trees decision trees and gradient boosting won't accept anymore `minsampleleavesplit=1` as at least 2 samples are required to split a decision tree node By Arnaud Joly

- `ensembleVotingClassifier` now raises `NotFittedError` if `predict` transform or `predict_proba` are called on the nonfitted estimator by Sebastian Raschka
- Fix bug where `ensembleAdaBoostClassifier` and `ensembleAdaBoostRegressor` would perform poorly if the `randomstate` was fixed 7411 By Joel Nothman
- Fix bug in ensembles with randomization where the ensemble would not set `randomstate` on base estimators in a pipeline or similar nesting 7411 Note results for `ensembleBaggingClassifier` `ensembleBaggingRegressor` `ensembleAdaBoostClassifier` and `ensembleAdaBoostRegressor` will now differ from previous versions By Joel Nothman

Linear kernelized and related models

- Fixed incorrect gradient computation for `losssquaredepsiloninsensitive` in `linearmodelSGDClassifier` and `linearmodelSGDRegressor` 6764 By Wenhua Yang

- Fix bug in `linearmodelLogisticRegressionCV` where `resolverliblinear` did not accept `classweightsbalanced` 6817 By Tom Dupre la Tour

- Fix bug in `neighborsRadiusNeighborsClassifier` where an error occurred when there were outliers being labelled and a weight function specified 6902 By LeonieBorne
- Fix `linearmodelElasticNet` sparse decision function to match output with dense in the multioutput case

Decomposition manifold learning and clustering

- `decompositionRandomizedPCA` default number of iterations `power` is 4 instead of 3 5141 by

Giorgio Patrini

- `utilsextramatrandomizedsvd` performs 4 power iterations by default instead of 0 In practice this is enough for obtaining a good approximation of the true eigenvalues/vectors in the presence of noise When `ncomponents` is small `1/min(X.shape[0], X.shape[1])` `niter` is set to 7 unless the user specifies a higher number This improves precision with few components 5299 by Giorgio Patrini

- `WhitenNonWhiten` inconsistency between components of `decompositionPCA` and `decompositionRandomizedPCA` now factored into `PCA` (Principal Component Analysis) see the `New features` is fixed components are stored with no whitening 5299 by Giorgio Patrini

- Fixed bug in `manifoldspectralembedding` where diagonal of unnormalized Laplacian matrix was incorrectly set to 1 4995 by Peter Fischer

- Fixed incorrect initialization of `utils.sparsepack.eigsh` on all occurrences Affects `cluster_bicluster_spectral` `biclustering` `decompositionKernelPCA` `manifoldLocallyLinearEmbedding` and `manifoldSpectralEmbedding` 5012 By Peter Fischer

- `Attribute explained_variance_ratio` calculated with the SVD solver of `discriminantanalysis.LinearDiscriminantAnalysis` now returns correct results By JPFrancoia

Preprocessing and feature selection

- `preprocessing.data_transform_selected` now always passes a copy of `X` to transform function when `copy=True` 7194 By Caio Oliveira

scikitlearn user guide Release 0213

Model evaluation and metaestimators

- modelselectionStratifiedKFold now raises error if all nlabels for individual classes is less than nfolds 6182 by Devashish Deshpande
- Fixed bug in modelselectionStratifiedShuffleSplit where train and test sample could overlap in some edge cases see 6121 for more details By Loic Esteve
- Fix in sklearnmodelselectionStratifiedShuffleSplit to return splits of size trainsize andtestsize in all cases 6472 By Andreas Müller
- Crossvalidation of OneVsOneClassifier andOneVsRestClassifier now works with precomputed kernels 7350 by Russell Smith
- Fix incomplete predictproba method delegation from modelselectionGridSearchCV to linearmodelSGDClassifier 7159 by Yichuan Liu

Metrics

- Fix bug in metricssilhouettescore in which clusters of size 1 were incorrectly scored They should get a score of 0 By Joel Nothman
- Fix bug in metricssilhouettesamples so that it now works with arbitrary labels not just those ranging from 0 to nclusters 1
- Fix bug where expected and adjusted mutual information were incorrect if cluster contingency cells exceeded 216 By Joel Nothman
- metricspairwisepairwisedistances now converts arrays to boolean arrays when required in scipyspatialdistance 5460 by Tom Dupre la Tour
- Fix sparse input support in metricssilhouettescore as well as example exam
- plestextdocumentclusteringpy By YenChen Lin
- metricsroccurve andmetricsprecisionrecallcurve no longer round yscore values when creating ROC curves this was causing problems for users with very small differences in scores 7353

Miscellaneous

- modelselectiontestsearchcheckparamgrid now works correctly with all types that extendsimplements Sequence except string including range Python 3x and xrange Python 2x 7323 by Viacheslav Kovalevskyi
 - utilsextmathrandomizedrangefinder is more numerically stable when many power iterations are requested since it applies LU normalization by default If niter2 numerical issues are unlikely thus no normalization is applied Other normalization options are available none LU andQR 5141 by Giorgio Patrini
 - Fix a bug where some formats of scipysparse matrix and estimators with them as parameters could not be passed to baseclone By Loic Esteve
 - datasetsloadsvmlightfile now is able to read long int QID values 7101 by Ibraim Ganiev
- API (Application Programming Interface) changes summary
- Linear kernelized and related models
- residualmetric has been deprecated in linearmodelIRANSACRegressor Use loss instead
- By Manoj Kumar
- Access to public attributes X and y has been deprecated in isotonicIsotonicRegression By Jonathan Arfa

scikitlearn user guide Release 0213

Decomposition manifold learning and clustering

- The oldmixtureDPGMM is deprecated in favor of the new mixtureBayesianGaussianMixture with the parameter weightconcentrationpriortypedirichletprocess The new class solves the computational problems of the old class and computes the Gaussian mixture with a Dirichlet process prior faster than before 7295 by Wei Xue and Thierry Guillemot
- The oldmixtureVBGMM is deprecated in favor of the new mixtureBayesianGaussianMixture with the parameter weightconcentrationpriortypedirichletdistribution The new class solves the computational problems of the old class and computes the Variational Bayesian Gaussian mixture faster than before 6651 by Wei Xue and Thierry Guillemot
- The oldmixtureGMM is deprecated in favor of the new mixtureGaussianMixture The new class computes the Gaussian mixture faster than before and some of computational problems have been solved 6666 by Wei Xue and Thierry Guillemot

Model evaluation and metaestimators

- Thesklearncrossvalidation sklearngridsearch andsklearnlearningcurve have been deprecated and the classes and functions have been reorganized into the sklearn modelselection module Ref Model Selection Enhancements and API (Application Programming Interface) Changes for more information 4294 by Raghav RV
- Thegridscores attribute of modelselectionGridSearchCV andmodelselection RandomizedSearchCV is deprecated in favor of the attribute cvresults Ref Model Selection Enhancements and API (Application Programming Interface) Changes for more information 6697 by Raghav RV
- The parameters niter ornolds in old CV (Cross-Validation) splitters are replaced by the new parameter nsplits since it can provide a consistent and unambiguous interface to represent the number of traintest splits 7187 by YenChen Lin
- classes parameter was renamed to labels inmetricshammingloss 7260 by Sebastián Vanrell
- The splitter classes LabelKFold LabelShuffleSplit LeaveOneLabelOut and LeavePLabelsOut are renamed to modelselectionGroupKFold modelselection GroupShuffleSplit modelselectionLeaveOneGroupOut andmodelselection LeavePGroupsOut respectively Also the parameter labels in thesplit method of the newly renamed splittersmodelselectionLeaveOneGroupOut andmodelselectionLeavePGroupsOut is renamed to groups Additionally in modelselectionLeavePGroupsOut the parameter nlabels is renamed to ngroups 6660 by Raghav RV
- Error and loss names for scoring parameters are now prefixed by neg such as negmeansquarederror The unprefixed versions are deprecated and will be removed in version 020 7261 by Tim Head

Code Contributors

Aditya Joshi Alejandro Alexander Fabisch Alexander Loginov Alexander Minyushkin Alexander Rudy Alexandre Abadie Alexandre Abraham Alexandre Gramfort Alexandre Saint alexfields Alvaro Ulloa alyssaq Amlan Kar Andreas Mueller andrew giessel Andrew Jackson Andrew McCulloh Andrew Murray Anish Shah Arafat Archit Sharma Ariel Rokem Arnaud Joly Arnaud Rachez Arthur Mensch Ash Hoover asnt b0nol Behzad Tabib ian Bernardo Bernhard Kratzwald Bhargav Mangipudi blakeflel Boyuan Deng Brandon Carter Brett Naul Brian McFee Caio Oliveira Camilo Lamus Carol Willing Cass CeShine Lee Charles Truong ChyiKwei Yau CJ Carey codevig Colin Ni Dan Shiebler Daniel Daniel Hnyk David Ellis David Nicholson David Staub David Thaler David Warshaw Davide Lasagna Deborah definitelyuncertain Didi BarZev djipey dsquareindia edwinENSAE Elias Kuthe Elvis DOHMATOB Ethan White Fabian Pedregosa Fabio Ticconi fisache Florian Wilhelm Francis Francis O'Donovan Gael Varoquaux Ganiev Ibraim ghg Gilles Louppe Giorgio Patrini Giovanni Cherubin Giovanni Lanzani Glenn Qian Gordon Mohr govinvatsan Graham Clenaghan Greg Reda Greg Stupp Guillaume 86 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

Lemaitre Gustav Mörtberg halwai Harizo Rajaona Harry Mavroforakis hashcode55 hdmeter Henry Lin Hobson Lane Hugo BowneAnderson Igor Andriushchenko Imaculate Inki Hwang Isaac Sijaranamual Ishank Gulati Issam Laradji Iver Jordal jackmartin Jacob Schreiber Jake Vanderplas James Fiedler James Routley Jan Zikes Janna Brettingen jarfa Jason Laska jblackburne jeff levesque Jeffrey Blackburne Jeffrey04 Jeremy Hintz jere mynixon Jeroen Jessica Yung JillJënn Vie Jimmy Jia Jiyuan Qian Joel Nothman johannah John John Boersma John Kirkham John Moeller jonathanstriebe1 joncrall Jordi Joseph Munoz Joshua Cook JPFrancoia jrfiedler JulianKahnert juliathebrave kaichogami KamalakerDadi Kenneth Lyons Kevin Wang kingjr kjell Konstantin Podshumok Kornel Kielczewski Krishna Kalyan krishnakalyan3 Kvie Putnam Kyle Jackson Lars Buitinck Idavid LeiG LeightonZhang Leland McInnes LiangChi Hsieh Lilian Besson lizzsz Loic Esteve Louis Tiao Léonie Borne Mads Jensen Maniteja Nandana Manoj Kumar Manvendra Singh Marco Mario Krell Mark Bao Mark Szepleniec Martin Madsen MartinBpr MaryanMorel Massil Matheus Mathieu Blondel Mathieu Dubois Matteo Matthias Ekman Max Moroz Michael Scherer michiaki ariga Mikhail Korobov Moussa Taifi mrandrewandrade Mridul Seth nadyap Naoya Kanai Nate George Nelle Varoquaux Nelson Liu Nick James NickleDave Nico Nicolas Goix Nikolay Mayorov ningchi nlathia okbalefthanded Okhlopkov Olivier Grisel Panos Louridas Paul Strickland Per rine Letellier pestrickland Peter Fischer Pieter PingYao Chang practicalswift Preston Parry Qimu Zheng Rachit Kansal Raghav RV Ralf Gommers RamanaS Rammig Randy Olson Rob Alexander Robert Lutz Robin Schucker Rohan Jain Ruifeng Zheng Ryan Yu Rémy Léone saihitam Saiwing Yeung Sam Shleifer Samuel StJean Sar taj Singh Sasank Chilamkurthy saurabhkansod Scott Andrews Scott Lowe seales Sebastian Raschka Sebastian Saeger Sebastián Vanrell Sergei Lebedev shagun Sodhani shanmuga cv Shashank Shekhar shawpan shengxid uan Shota shuckle16 Skipper Seabold sklearnci SmedbergM srvanrell Sébastien Lerique Taranjeet themrmax Thierry Thierry Guillemot Thomas Thomas Hallock Thomas Moreau Tim Head tKammy toastedcornflakes Tom TomDLT Toshihiro Kamishima tracer0tong Trent Hauck trevorstephens Tue V o Varun Varun Jewalikar Viach eslav Vighnesh Birodkar Vikram Villu Ruusmann Vinayak Mehta walter waterponey Wenhua Yang Wenjian Huang Will Welch wyseguy7 xyguo yanlend Yaroslav Halchenko yelite Yen YenChenLin Yichuan Liu Yoav Ram Yoshiki Zheng RuiFeng zivori Óscar Nájera

1177 Version 0171

February 18 2016

Changelog

Bug fixes

- Upgrade vendored joblib to version 094 that fixes an important bug in joblibParallel that can silently yield to wrong results when working on datasets larger than 1MB <https://github.com/joblib/joblib/blob/094>

CHANGESrst

- Fixed reading of Bunch pickles generated with scikitlearn version 016 This can affect users who have already downloaded a dataset with scikitlearn 016 and are loading it with scikitlearn 017 See 6196 for how this affected datasetsfetch20newsgroups By Loic Esteve

- Fixed a bug that prevented using ROC AUC score to perform grid search on several CPU cores on large arrays See 6147 By Olivier Grisel

- Fixed a bug that prevented to properly set the presort parameter in ensemble GradientBoostingRegressor See 5857 By Andrew McCulloh

- Fixed a joblib error when evaluating the perplexity of a decomposition LatentDirichletAllocation model See 6258 By ChyiKwei Yau

117 Previous Releases 87

scikitlearn user guide Release 0213

1178 Version 017

November 5 2015

Changelog

New features

- All the Scaler classes but preprocessingRobustScaler can be fitted online by calling partialfit By Giorgio Patrini
 - The new class ensembleVotingClassifier implements a “majority rule” “soft voting” ensemble classifier to combine estimators for classification By Sebastian Raschka
 - The new class preprocessingRobustScaler provides an alternative to preprocessingStandardScaler for featurewise centering and range normalization that is robust to outliers By Thomas Unterthiner
 - The new class preprocessingMaxAbsScaler provides an alternative to preprocessingMinMaxScaler for featurewise range normalization when the data is already centered or sparse By Thomas Unterthiner
 - The new class preprocessingFunctionTransformer turns a Python function into a Pipeline compatible transformer object By Joe Jevnik
 - The new classes crossvalidationLabelKFold andcrossvalidationLabelShuffleSplit generate traintest folds respectively similar to crossvalidationKFold and crossvalidationShuffleSplit except that the folds are conditioned on a label array By Brian McFee Jean Kossaifi and Gilles Louppe
 - decompositionLatentDirichletAllocation implements the Latent Dirichlet Allocation topic model with online variational inference By ChyiKwei Yau with code based on an implementation by Matt Hoffman 3659
 - The new solver sag implements a Stochastic Average Gradient descent and is available in both linearmodelLogisticRegression andlinearmodelRidge This solver is very efficient for large datasets By Danny Sullivan and Tom Dupre la Tour 4738
 - The new solver cdimplements a Coordinate Descent in decompositionNMF Previous solver based on Projected Gradient is still available setting new parameter solver topg but is deprecated and will be removed in 019 along with decompositionProjectedGradientNMF and parameters sparseness eta beta andnlsmxiter New parameters alpha andl1ratio control L1 and L2 regularization and shuffle adds a shuffling step in the cdsolver By Tom Dupre la Tour and Mathieu Blondel
- Enhancements
- manifoldTSNE now supports approximate optimization via the BarnesHut method leading to much faster fitting By Christopher Erick Moody 4025
 - clustermeanshiftMeanShift now supports parallel execution as implemented in the meanshift function By Martino Sorbaro
 - naivebayesGaussianNB now supports fitting with sampleweight By Jan Hendrik Metzen
 - dummyDummyClassifier now supports a prior fitting strategy By Arnaud Joly
 - Added afitpredict method for mixtureGMM and subclasses By Cory Lorenz

88 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Added the metricslabelrankingloss metric By Arnaud Joly
- Added the metricscohenkappascore metric
- Added a warmstart constructor parameter to the bagging ensemble models to increase the size of the ensemble By Tim Head
- Added option to use multioutput regression metrics without averaging By Konstantin Shmelkov and Michael Eickenberg
- Addedstratify option tocrossvalidationtraintestsplit for stratified splitting By Miroslav Batchkarov
- Thetreeexportgraphviz function now supports aesthetic improvements for tree DecisionTreeClassifier andtreeDecisionTreeRegressor including options for coloring nodes by their majority class or impurity showing variable names and using node proportions instead of raw sample counts By Trevor Stephens
- Improved speed of newtoncg solver inlinearmodelLogisticRegression by avoiding loss computation By Mathieu Blondel and Tom Dupre la Tour
- Theclassweightauto heuristic in classifiers supporting classweight was deprecated and replaced by the classweightbalanced option which has a simpler formula and interpretation By Hanna Wallach and Andreas Müller
- Addclassweight parameter to automatically weight samples by class frequency for linearmodel PassiveAggressiveClassifier By Trevor Stephens
- Added backlinks from the API (Application Programming Interface) reference pages to the user guide By Andreas Müller
- Thelabels parameter to sklearnmetricsf1score sklearnmetricsfbetascore sklearnmetricsrecallscore andsklearnmetricsprecisionscore has been extended It is now possible to ignore one or more labels such as where a multiclass problem has a majority class to ignore By Joel Nothman
- Addsampleweight support tolinearmodelRidgeClassifier By Trevor Stephens
- Provide an option for sparse output from sklearnmetricspairwisecosinesimilarity By Jaidev Deshpande
- Addminmaxscale to provide a function interface for MinMaxScaler By Thomas Unterthiner
- dumpsvmlightfile now handles multilabel datasets By ChihWei Chang
- RCV1 dataset loader sklearndatasetsfetchrcv1 By Tom Dupre la Tour
- The “Wisconsin Breast Cancer” classical twoclass classification dataset is now included in scikitlearn available withsklearndatasetloadbreastcancer
- Upgraded to joblib 093 to benefit from the new automatic batching of short tasks This makes it possible for scikitlearn to benefit from parallelism when many very short tasks are executed in parallel for instance by the gridsearchGridSearchCV metaestimator with njobs 1 used with a large grid of parameters on a small dataset By Vlad Niculae Olivier Grisel and Loic Esteve
- For more details about changes in joblib 093 see the release notes <https://github.com/joblib/joblib/blob/master/CHANGESrstrelease093>
- Improved speed 3 times per iteration of decompositionDictLearning with coordinate descent method from linearmodelLasso By Arthur Mensch
- Parallel processing threaded for queries of nearest neighbors using the balltree by Nikolay Mayorov
- Allowdatasetmakemultilabelclassification to output a sparse y By Kashif Rasul

scikitlearn user guide Release 0213

- clusterDBSCAN now accepts a sparse matrix of precomputed distances allowing memoryefficient distance precomputation By Joel Nothman
 - treeDecisionTreeClassifier now exposes an apply method for retrieving the leaf indices samples are predicted as By Daniel Galvez and Gilles Louppe
 - Speed up decision tree regressors random forest regressors extra trees regressors and gradient boosting estimators by computing a proxy of the impurity improvement during the tree growth The proxy quantity is such that the split that maximizes this value also maximizes the impurity improvement By Arnaud Joly Jacob Schreiber and Gilles Louppe
 - Speed up tree based methods by reducing the number of computations needed when computing the impurity measure taking into account linear relationship of the computed statistics The effect is particularly visible with extra trees and on datasets with categorical or sparse features By Arnaud Joly
 - ensembleGradientBoostingRegressor andensembleGradientBoostingClassifier now expose an apply method for retrieving the leaf indices each sample ends up in under each try By Jacob Schreiber
 - Addsampleweight support tolinearmodelLinearRegression By Sonny Hu 4881
 - Addniterwithoutprogress tomanifoldTSNE to control the stopping criterion By Santi Vialba 5186
 - Added optional parameter randomstate inlinearmodelRidge to set the seed of the pseudo random generator used in sag solver By Tom Dupre la Tour
 - Added optional parameter warmstart inlinearmodelLogisticRegression If set to True the solverslbfgs newtoncg andsag will be initialized with the coefficients computed in the previous fit By Tom Dupre la Tour
 - Addedsampleweight support to linearmodelLogisticRegression for thelbfgs newtoncg andsag solvers By Valentin Stolbunov Support added to the liblinear solver By Manoj Kumar
 - Added optional parameter presort toensembleGradientBoostingRegressor andensemble GradientBoostingClassifier keeping default behavior the same This allows gradient boosters to turn off presorting when building deep trees or using sparse data By Jacob Schreiber
 - Alteredmetricsroccurve to drop unnecessary thresholds by default By Graham Clenaghan
 - AddedfeatureselectionSelectFromModel metatransformer which can be used along with estimators that have coef orfeatureimportances attribute to select important features of the input data By Maheshakya Wijewardena Joel Nothman and Manoj Kumar
 - Addedmetricspairwiselaplaciankernel By Clyde Fare
 - covarianceGraphLasso allows separate control of the convergence criterion for the ElasticNet subproblem via the enettol parameter
 - Improved verbosity in decompositionDictionaryLearning
 - ensembleRandomForestClassifier andensembleRandomForestRegressor no longer explicitly store the samples used in bagging resulting in a much reduced memory footprint for storing random forest models
 - Addedpositive option tolinearmodelLars andlinearmodelLarsPath to force coefficients to be positive 5131
 - Added the Xnormsquared parameter to metricspairwiseeuclideananddistances to provide precomputed squared norms for X
 - Added the fitpredict method topipelinePipeline
- 90 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Added the preprocessingminmaxscale function

Bug fixes

- Fixed nondeterminism in dummyDummyClassifier with sparse multilabel output By Andreas Müller
- Fixed the output shape of linearmodelRANSACRegressor tonsamples By Andreas Müller
- Fixed bug in decompositionDictLearning when njobs = 0 By Andreas Müller
- Fixed bug where gridsearchRandomizedSearchCV could consume a lot of memory for large discrete grids By Joel Nothman
- Fixed bug in linearmodelLogisticRegressionCV where penalty was ignored in the final fit By Manoj Kumar
- Fixed bug in ensembleforestForestClassifier while computing oobscore and X is a sparse matrix By Ankur Ankan
- All regressors now consistently handle and warn when given y that is of shape n samples 1 By Andreas Müller and Henry Lin 5431
- Fix in clusterKMeans cluster reassignment for sparse input by Lars Buitinck
- Fixed a bug in lassoLDA that could cause asymmetric covariance matrices when using shrinkage By Martin Billinger
- Fixed crossvalidationcrossvalpredict for estimators with sparse predictions By Buddha Prakash
- Fixed the predict_proba method of linearmodelLogisticRegression to use softmax instead of onevsrest normalization By Manoj Kumar 5182
- Fixed the partial_fit method of linearmodelSGDClassifier when called with average=True By Andrew Lamb 5282
- Dataset fetchers use different filenames under Python 2 and Python 3 to avoid pickling compatibility issues By Olivier Grisel 5355
- Fixed a bug in naivebayesGaussianNB which caused classification results to depend on scale By Jake Vanderplas
- Fixed temporarily linearmodelRidge which was incorrect when fitting the intercept in the case of sparse data The fix automatically changes the solver to 'sag' in this case 5360 by Tom Dupre la Tour
- Fixed a performance bug in decompositionRandomizedPCA on data with a large number of features and fewer samples 4478 By Andreas Müller Loic Esteve and Giorgio Patrini
- Fixed bug in crossdecompositionPLS that yielded unstable and platform dependent output and failed on fit_transform By Arthur Mensch
- Fixes to the Bunch class used to store datasets
- Fixed ensembleplotpartialdependence ignoring the percentiles parameter
- Providing a set as vocabulary in CountVectorizer no longer leads to inconsistent results when pickling
- Fixed the conditions on when a precomputed Gram matrix needs to be recomputed in linearmodelLinearRegression linearmodelOrthogonalMatchingPursuit linearmodelLasso and linearmodelElasticNet
- Fixed inconsistent memory layout in the coordinate descent solver that affected linearmodelDictionaryLearning and covarianceGraphLasso 5337 By Olivier Grisel

117 Previous Releases 91

scikitlearn user guide Release 0213

- manifoldLocallyLinearEmbedding no longer ignores the reg parameter
- Nearest Neighbor estimators with custom distance metrics can now be pickled 4362
- Fixed a bug in pipelineFeatureUnion wheretransformerweights were not properly handled when performing gridsearches
- Fixed a bug in linearmodelLogisticRegression andlinearmodelLogisticRegressionCV when using classweightbalanced orclassweightauto

By Tom Dupre la Tour

- Fixed bug 5495 when doing OVRSVCDdecisionfunctionsshape"ovr" Fixed by Elvis Dohmatob
- API (Application Programming Interface) changes summary
- Attribute datamin datamax anddatarange inpreprocessingMinMaxScaler are deprecated and won't be available from 019 Instead the class now exposes datamin datamax and datarange By Giorgio Patrini
 - All Scaler classes now have an scale attribute the featurewise rescaling applied by their transform methods The old attribute std inpreprocessingStandardScaler is deprecated and superseded by scale it won't be available in 019 By Giorgio Patrini
 - svmSVC andsvmNuSVC now have an decisionfunctionsshape parameter to make their decision function of shape nsamples nclasses by settingdecisionfunctionsshapeovr This will be the default behavior starting in 019 By Andreas Müller
 - Passing 1D data arrays as input to estimators is now deprecated as it caused confusion in how the array elements should be interpreted as features or as samples All data arrays are now expected to be explicitly shaped nsamples nfeatures By Vighnesh Birodkar
 - ldaLDA andqdaQDA have been moved to discriminantanalysis LinearDiscriminantAnalysis and discriminantanalysis QuadraticDiscriminantAnalysis
 - Thestorecovariance andtol parameters have been moved from the fit method to the constructor in discriminantanalysisLinearDiscriminantAnalysis and thestorecovariances and tol parameters have been moved from the fit method to the constructor in discriminantanalysis QuadraticDiscriminantAnalysis
 - Models inheriting from LearntSelectorMixin will no longer support the transform methods ie RandomForests GradientBoosting LogisticRegression DecisionTrees SVMs and SGD related models Wrap these models around the metatransformer featureselectionSelectFromModel to remove features according to coefs orfeatureimportances which are below a certain threshold value instead
 - clusterKMeans reruns clusterassignments in case of nonconvergence to ensure consistency of predictX andlabels By Vighnesh Birodkar
 - Classifier and Regressor models are now tagged as such using the estimatortype attribute
 - Crossvalidation iterators always provide indices into training and test set not boolean masks
 - Thedecisionfunction on all regressors was deprecated and will be removed in 019 Use predict instead
 - datasetsloadlfwpairs is deprecated and will be removed in 019 Use datasets fetchlfwpairs instead
 - The deprecated hmm module was removed
 - The deprecated Bootstrap crossvalidation iterator was removed

scikitlearn user guide Release 0213

- The deprecated Ward andWardAgglomerative classes have been removed Use clustering AgglomerativeClustering instead
- crossvalidationcheckcv is now a public function
- The property residues oflinearmodelLinearRegression is deprecated and will be removed in 019
- The deprecated njobs parameter of linearmodelLinearRegression has been moved to the constructor
- Removed deprecated classweight parameter from linearmodelSGDClassifier 'sfit method Use the construction parameter instead
- The deprecated support for the sequence of sequences or list of lists multilabel format was removed To convert to and from the supported binary indicator matrix format use MultiLabelBinarizer
- The behavior of calling the inversetransform method ofPipelinepipeline will change in 019 It will no longer reshape onedimensional input to twodimensional input
- The deprecated attributes indicatormatrix multilabel andclasses ofpreprocessing LabelBinarizer were removed
- Usinggamma0 insvmSVC andsvmSVR to automatically set the gamma to 1 nfeatures is deprecated and will be removed in 019 Use gammaauto instead

Code Contributors

Aaron Schumacher Adithya Ganesh akitty Alexandre Gramfort Alexey Grigorev Ali Baharev Allen Riddell Ando Saabas Andreas Mueller Andrew Lamb Anish Shah Ankur Ankan Anthony Erlinger Ari Rouvinen Arnaud Joly Arnaud Rachez Arthur Mensch banilo Barmaleyexe benjaminirving Boyuan Deng Brett Naul Brian McFee Buddha Prakash Chi Zhang ChihWei Chang Christof Angermueller Christoph Gohlke Christophe Bourguignat Christopher Erick Moody ChyiKwei Yau Cindy Sridharan CJ Carey Clydefare Cory Lorenz Dan Blanchard Daniel Galvez Daniel Kronovet Danny Sullivan Data1010 David David D Lowe David Dotson djiipay Dmitry Spikhalskiy Donne Martin Dougal J Sutherland Dougal Sutherland edson duarte Eduardo Caro Eric Larson Eric Martin Erich Schubert Fernando Carrillo Frank C Eckert Frank Zalkow Gael Varoquaux Ganiev Ibraim Gilles Louppe Giorgio Patrini giorgiop Graham Clenaghan Gryllos Prokopis gwulfs Henry Lin HsuanTien Lin Immanuel Bayer Ishank Gulati Jack Martin Jacob Schreiber Jaidev Deshpande Jake Vanderplas Jan Hendrik Metzen Jean Kossaifi Jeffrey04 Jeremy jfrac Jiali Mei Joe Jevnik Joel Nothman John Kirkham John Wittenauer Joseph Joshua Loyal Jungkook Park KamalakerDadi Kashif Rasul Keith Goodman Kian Ho Konstantin Shmelkov Kyler Brown Lars Buitinck Lilian Besson Loic Esteve Louis Tiao maheshakya Maheshakya Wijewardena Manoj Kumar MarkTab marktabnet Martin Ku Martin Spacek MartinBpr martinossorb MaryanMorel Masafumi Oyamada Mathieu Blondel Matt Krump Matti Lyra Maxim Kolganov mbillinger mhg Michael Heilman Michael Patterson Miroslav Batchkarov Nelle Varoquaux Nicolas Nikolay Mayorov Olivier Grisel Omer Katz Óscar Nájera Pauli Virtanen Peter Fischer Peter Prettenhofer Phil Roth pianomania Preston Parry Raghav RV Rob Zinkov Robert Layton Rohan Ramanath Saket Choudhary Sam Zhang santi saurabhbansod sclsl9fr Sebastian Raschka Sebastian Saeger Shivan Sornarajah SimonPL sinhrks Skipper Seabold Sonny Hu sseg Stephen Hoover Steven DeGryze Steven Seguin Theodore Vasiloudis Thomas Unterthiner Tiago Freitas Pereira Tian Wang Tim Head Timothy Hopper tokoroten Tom Dupré la Tour Trevor Stephens Valentin Stolbunov Vighnesh Birodkar Vinayak Mehta Vincent Vincent Michel vstolbunov wangz10 Wei Xue Yucheng Low Yury Zhauniarovich Zac Stewart zhaipro Zichen Wang

1179 Version 0161

April 14 2015

117 Previous Releases 93

scikitlearn user guide Release 0213

Changelog

Bug fixes

- Allow input data larger than blocksize in `covarianceLedoitWolf` by Andreas Müller
- Fix a bug in `isotonicIsotonicRegression` deduplication that caused unstable result in `calibrationCalibratedClassifierCV` by Jan Hendrik Metzen
- Fix sorting of labels in `func preprocessingLabelBinarize` by Michael Heilman
- Fix several stability and convergence issues in `crossdecompositionCCA` and `crossdecompositionPLSCanonical` by Andreas Müller
- Fix a bug in `clusterKMeans` when `precomputedDistances=False` on fortran ordered data
- Fix a speed regression in `ensembleRandomForestClassifier` 's `predict` and `predict_proba` by Andreas Müller
- Fix a regression where `utilsshuffle` converted lists and dataframes to arrays by Olivier Grisel

11710 Version 016

March 26 2015

Highlights

- Speed improvements notably in `clusterDBSCAN` reduced memory requirements bugfixes and better default settings
- Multinomial Logistic regression and a path algorithm in `linearModelLogisticRegressionCV`
- Out of core learning of PCA (Principal Component Analysis) via `decompositionIncrementalPCA`
- Probability calibration of classifiers using `calibrationCalibratedClassifierCV`
- `clusterBirch` clustering method for large scale datasets
- Scalable approximate nearest neighbors search with Locality sensitive hashing forests in `neighborsLSHForest`
- Improved error messages and better validation when using malformed input data
- More robust integration with pandas dataframes

Changelog

New features

- The new `neighborsLSHForest` implements locality sensitive hashing for approximate nearest neighbors search By Maheshakya Wijewardena
- Added `svmLinearSVR` This class uses the liblinear implementation of Support Vector Regression which is much faster for large sample sizes than `svmSVR` with linear kernel By Fabian Pedregosa and Qiang Luo
- Incremental fit for `GaussianNB`
- Added `sample_weight` support to `dummyDummyClassifier` and `dummyDummyRegressor` By Arnaud Joly

scikitlearn user guide Release 0213

- Added the metricslabelrankingaverageprecisionscore metrics By Arnaud Joly
 - Add themetricscoverageerror metrics By Arnaud Joly
 - AddedlinearmodelLogisticRegressionCV By Manoj Kumar Fabian Pedregosa Gael Varoquaux and Alexandre Gramfort
 - Addedwarmstart constructor parameter to make it possible for any trained forest model to grow additional trees incrementally By Laurent Direr
 - Addedsampleweight support to ensembleGradientBoostingClassifier andensemble GradientBoostingRegressor By Peter Prettenhofer
 - AddeddecompositionIncrementalPCA an implementation of the PCA (Principal Component Analysis) algorithm that supports out ofcore learning with a partialfit method By Kyle Kastner
 - Averaged SGD for SGDClassifier andSGDRegressor By Danny Sullivan
 - Addedcrossvalpredict function which computes crossvalidated estimates By Luis Pedro Coelho
 - AddedlinearmodelTheilSenRegressor a robust generalizedmedianbased estimator By Florian Wilhelm
 - Addedmetricsmedianabsoluteerror a robust metric By Gael Varoquaux and Florian Wilhelm
 - AddedclusterBirch an online clustering algorithm By Manoj Kumar Alexandre Gramfort and Joel Nothman
 - Added shrinkage support to discriminantanalysisLinearDiscriminantAnalysis using two new solvers By Clemens Brunner and Martin Billinger
 - AddedkernelridgeKernelRidge an implementation of kernelized ridge regression By Mathieu Blondel and Jan Hendrik Metzen
 - All solvers in linearmodelRidge now support sampleweight By Mathieu Blondel
 - AddedcrossvalidationPredefinedSplit crossvalidation for fixed userprovided crossvalidation folds By Thomas Unterthiner
 - AddedcalibrationCalibratedClassifierCV an approach for calibrating the predicted probabilities of a classifier By Alexandre Gramfort Jan Hendrik Metzen Mathieu Blondel and Balazs Kegl
- Enhancements
- Add option returndistance inhierarchicalwardtree to return distances between nodes for both structured and unstructured versions of the algorithm By Matteo Visconti di Oleggio Castello The same option was added in hierarchicallinkagetree By Manoj Kumar
 - Add support for sample weights in scorer objects Metrics with sample weight support will automatically benefit from it By Noel Dawe and Vlad Niculae
 - Addednewtoncg andlbfgs solver support in linearmodelLogisticRegression By Manoj Kumar
 - Addselectionrandom parameter to implement stochastic coordinate descent for linearmodel Lasso linearmodelElasticNet and related By Manoj Kumar
 - Addsampleweight parameter to metricsjaccardsimilarityscore andmetrics logloss By Jatin Shah
 - Support sparse multilabel indicator representation in preprocessingLabelBinarizer and multiclassOneVsRestClassifier by Hamzeh Alsalhi with thanks to Rohit Sivaprasad as well as evaluation metrics by Joel Nothman
- 117 Previous Releases 95

scikitlearn user guide Release 0213

- Addsampleweight parameter to metricsjaccardsimilarityscore ByJatin Shah
 - Add support for multiclass in metricshingeloss AddedlabelsNone as optional parameter By Saurabh Jha
 - Addsampleweight parameter to metricshingeloss BySaurabh Jha
 - Addmulticlassmultinomial option inlinearmodelLogisticRegression to implement a Logistic Regression solver that minimizes the crossentropy or multinomial loss instead of the default OnevsRest setting Supports lbfgs andnewtoncg solvers By Lars Buitinck and Manoj Kumar Solver optionnewtoncg by Simon Wu
 - DictVectorizer can now perform fittransform on an iterable in a single pass when giving the option sortFalse By Dan Blanchard
 - GridSearchCV andRandomizedSearchCV can now be configured to work with estimators that may fail and raise errors on individual folds This option is controlled by the errorscore parameter This does not affect errors raised on refit By Michal Romaniuk
 - Adddigits parameter to metricsclassificationreport to allow report to show different precision of floating point numbers By Ian Gilmore
 - Add a quantile prediction strategy to the dummyDummyRegressor By Aaron Staple
 - Addhandleunknown option to preprocessingOneHotEncoder to handle unknown categorical features more gracefully during transform By Manoj Kumar
 - Added support for sparse input data to decision trees and their ensembles By Fares Hedyati and Arnaud Joly
 - Optimized clusterAffinityPropagation by reducing the number of memory allocations of large temporary datastructures By Antony Lee
 - Parallelization of the computation of feature importances in random forest By Olivier Grisel and Arnaud Joly
 - Addniter attribute to estimators that accept a maxiter attribute in their constructor By Manoj Kumar
 - Added decision function for multiclassOneVsOneClassifier By Raghav RV and Kyle Beauchamp
 - neighborskneighborsgraph andradiusneighborsgraph support nonEuclidean metrics By Manoj Kumar
 - Parameter connectivity inclusterAgglomerativeClustering and family now accept callables that return a connectivity matrix By Manoj Kumar
 - Sparse support for paireddistances By Joel Nothman
 - clusterDBSCAN now supports sparse input and sample weights and has been optimized the inner loop has been rewritten in Cython and radius neighbors queries are now computed in batch By Joel Nothman and Lars Buitinck
 - Addclassweight parameter to automatically weight samples by class frequency for ensembleRandomForestClassifier treeDecisionTreeClassifier ensemble ExtraTreesClassifier andtreeExtraTreeClassifier By Trevor Stephens
 - gridsearchRandomizedSearchCV now does sampling without replacement if all parameters are given as lists By Andreas Müller
 - Parallelized calculation of pairwisedistances is now supported for scipy metrics and custom callables By Joel Nothman
 - Allow the fitting and scoring of all clustering algorithms in pipelinePipeline By Andreas Müller
 - More robust seeding and improved error messages in clusterMeanShift by Andreas Müller
- 96 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Make the stopping criterion for mixtureGMM mixtureDPGMM andmixtureVBGMM less dependent on the number of samples by thresholding the average loglikelihood change instead of its sum over all samples By Hervé Bredin

- The outcome of manifoldspectralembedding was made deterministic by flipping the sign of eigen vectors By Hasil Sharma

- Significant performance and memory usage improvements in preprocessingPolynomialFeatures By Eric Martin

- Numerical stability improvements for preprocessingStandardScaler andpreprocessing scale By Nicolas Goix

- svmSVC fitted on sparse input now implements decisionfunction By Rob Zinkov and Andreas Müller

- crossvalidationtraintestsplit now preserves the input type instead of converting to numpy arrays

Documentation improvements

- Added example of using FeatureUnion for heterogeneous input By Matt Terry

- Documentation on scorers was improved to highlight the handling of loss functions By Matt Pico

- A discrepancy between liblinear output and scikitlearn’s wrappers is now noted By Manoj Kumar

- Improved documentation generation examples referring to a class or function are now shown in a gallery on the classfunction’s API (Application Programming Interface) reference page By Joel Nothman

- More explicit documentation of sample generators and of data transformation By Joel Nothman

- sklearnneighborsBallTree andsklearnneighborsKDTree used to point to empty pages

stating that they are aliases of BinaryTree This has been fixed to show the correct class docs By Manoj Kumar

- Added silhouette plots for analysis of KMeans clustering using metricssilhouettesamples and metricssilhouettescore See Selecting the number of clusters with silhouette analysis on KMeans clustering

Bug fixes

- Metaestimators now support ducktyping for the presence of decisionfunction

predictproba and other methods This fixes behavior of gridsearchGridSearchCV

gridsearchRandomizedSearchCV pipelinePipeline featuresselectionRFE

featuresselectionRFECV when nested By Joel Nothman

- Thescoring attribute of gridsearch and crossvalidation methods is no longer ignored when a

gridsearchGridSearchCV is given as a base estimator or the base estimator doesn’t have predict

- The function hierarchicalwardtree now returns the children in the same order for both the structured and unstructured versions By Matteo Visconti di Oleggio Castello

- featuresselectionRFECV now correctly handles cases when step is not equal to 1 By Nikolay Mayorov

- ThedecompositionPCA now undoes whitening in its inversetransform Also itscomponents now always have unit length By Michael Eickenberg

- Fix incomplete download of the dataset when datasetsdownload20newsgroups is called By Manoj Kumar

117 Previous Releases 97

scikitlearn user guide Release 0213

- Various fixes to the Gaussian processes subpackage by Vincent Dubourg and Jan Hendrik Metzen
- Callingpartialfit withclassweightauto throws an appropriate error message and suggests a work around By Danny Sullivan
- RBFSampler withgammag formerly approximated rbfkernel withgammag2 the definition of gamma is now consistent which may substantially change your results if you use a fixed value If you cross validated over gamma it probably doesn't matter too much By Dougal Sutherland
- Pipeline object delegate the classes attribute to the underlying estimator It allows for instance to make bagging of a pipeline object By Arnaud Joly
- neighborsNearestCentroid now uses the median as the centroid when metric is set to manhattan It was using the mean before By Manoj Kumar
- Fix numerical stability issues in linearmodelSGDClassifier andlinearmodelSGDRegressor by clipping large gradients and ensuring that weight decay rescaling is always positive for large l2 regularization and large learning rate values By Olivier Grisel
- Whencomputealltree is set to "auto" the full tree is built when nclusters is high and is early stopped when nclusters is low while the behavior should be viceversa in clusterAgglomerativeClustering and friends This has been fixed By Manoj Kumar
- Fix lazy centering of data in linearmodelenetpath andlinearmodellassopath It was centered around one It has been changed to be centered around the origin By Manoj Kumar
- Fix handling of precomputed affinity matrices in clusterAgglomerativeClustering when using connectivity constraints By Cathy Deng
- Correctpartialfit handling of classprior forsklearnnaivebayesMultinomialNB andsklearnnaivebayesBernoulliNB By Trevor Stephens
- Fixed a crash in metricsprecisionrecallfscoresupport when using unsorted labels in the multilabel setting By Andreas Müller
- Avoid skipping the first nearest neighbor in the methods radiusneighbors kneighbors kneighborsgraph andradiusneighborsgraph insklearnneighborsNearestNeighbors and family when the query data is not the same as fit data By Manoj Kumar
- Fix logdensity calculation in the mixtureGMM with tied covariance By Will Dawson
- Fixed a scaling error in featureselectionSelectFdr where a factor nfeatures was missing By Andrew Tulloch
- Fix zero division in neighborsKNeighborsRegressor and related classes when using distance weighting and having identical data points By GarretR
- Fixed round off errors with non positivedefinite covariance matrices in GMM By Alexis Mignon
- Fixed a error in the computation of conditional probabilities in naivebayesBernoulliNB By Hanna Wallach
- Make the method radiusneighbors ofneighborsNearestNeighbors return the samples lying on the boundary for algorithmbrute By Yan Yi
- Flip sign of dualcoef ofsvmSVC to make it consistent with the documentation and decisionfunction By Artem Sobolev
- Fixed handling of ties in isotonicIsotonicRegression We now use the weighted average of targets secondary method By Andreas Müller and Michael Bommarito

scikitlearn user guide Release 0213

API (Application Programming Interface) changes summary

- GridSearchCV andcrossvalscore and other metaestimators don't convert pandas DataFrames into arrays any more allowing DataFrame specific operations in custom estimators
- multiclassfitovr multiclasspredictovr predictprobaovr multiclassfitovo multiclasspredictovo multiclassfitecoc andmulticlasspredictecoc are deprecated Use the underlying estimators instead
- Nearest neighbors estimators used to take arbitrary keyword arguments and pass these to their distance metric

This will no longer be supported in scikitlearn 018 use the metricparams argument instead

- njobs parameter of the fit method shifted to the constructor of the LinearRegression class
- Thepredictproba method ofmulticlassOneVsRestClassifier now returns two probabilities per sample in the multiclass case this is consistent with other estimators and with the method's documentation but previous versions accidentally returned only the positive probability Fixed by Will Lamond and Lars Buitinck

- Change default value of precompute in ElasticNet andLasso to False Setting precompute to "auto" was found to be slower when nsamples nfeatures since the computation of the Gram matrix is computationally expensive and outweighs the benefit of fitting the Gram for just one alpha precomputeauto is now deprecated and will be removed in 018 By Manoj Kumar

- Exposepositive option inlinearmodelenetpath andlinearmodelenetpath which constrains coefficients to be positive By Manoj Kumar
- Users should now supply an explicit average parameter to sklearnmetricsf1score sklearnmetricsfbetascore sklearnmetricsrecallscore andsklearnmetricsprecisionscore when performing multiclass or multilabel ie not binary classification By Joel Nothman

- scoring parameter for cross validation now accepts f1micro f1macro orf1weighted f1 is now for binary classification only Similar changes apply to precision andrecall By Joel Nothman

- Thefitintercept normalize andreturnmodels parameters in linearmodelenetpath andlinearmodellassopath have been removed They were deprecated since 014

- From now onwards all estimators will uniformly raise NotFittedError utilsvalidation NotFittedError when any of the predict like methods are called before the model is fit By Raghav RV

- Input data validation was refactored for more consistent input validation The checkarrays function was replaced by checkarray andcheckXy By Andreas Müller

- AllowXNone in the methods radiusneighbors kneighbors kneighborsgraph and radiusneighborsgraph insklearnneighborsNearestNeighbors and family If set to None then for every sample this avoids setting the sample itself as the first nearest neighbor By Manoj Kumar

- Add parameter includeself inneighborskneighborsgraph andneighborsradiusneighborsgraph which has to be explicitly set by the user If set to True then the sample itself is considered as the first nearest neighbor

- thresh parameter is deprecated in favor of new tol parameter in GMDPGMM andVBGMM See Enhancements section for details By Hervé Bredin

- Estimators will treat input with dtype object as numeric when possible By Andreas Müller
- Estimators now raise ValueError consistently when fitted on empty data less than 1 sample or less than 1 feature for 2D input By Olivier Grisel

scikitlearn user guide Release 0213

- Theshuffle option of linearmodelSGDClassifier linearmodelSGDRegressor linearmodelPerceptron linearmodelPassiveAggressiveClassifier and linearmodelPassiveAggressiveRegressor now defaults to True
- clusterDBSCAN now uses a deterministic initialization The randomstate parameter is deprecated By Erich Schubert

Code Contributors

A Flaxman Aaron Schumacher Aaron Staple abhishek thakur Akshay akshayah3 Aldrian Obaja Alexander Fabisch Alexandre Gramfort Alexis Mignon Anders Aagaard Andreas Mueller Andreas van Cranenburgh Andrew Tulloch Andrew Walker Antony Lee Arnaud Joly banilo Barmaleyex Ben Davies Benedikt Koehler bhsu Boris Feld Borja Ayerdi Boyuan Deng Brent Pedersen Brian Wignall Brooke Osborn Calvin Giles Cathy Deng Celeo cgohlke chebee7i Christian Stadelhardt Christof Angermueller ChyiKwei Yau CJ Carey Clemens Brunner Daiki Aminaka Dan Blanchard danfrankj Danny Sullivan David Fletcher Dmitrijs Milajevs Dougal J Sutherland Erich Schubert Fabian Pedregosa Florian Wilhelm floydsoft FélixAntoine Fortin Gael Varoquaux GarrettR Gilles Louppe gpassino gwulfs Hampus Bengtsson Hamzeh Alsalhi Hanna Wallach Harry Mavroforakis Hasil Sharma Helder Herve Bredin HsiangFu Yu Hugues SALAMIN Ian Gilmore Ilambharathi Kanniah Imran Haque isms Jake VanderPlas Jan Dlabal Jan Hendrik Metzen Jatin Shah Javier López Peña jdcaballero Jean Kossaifi Jeff Hammerbacher Joel Nothman Jonathan Helmus Joseph Kaicheng Zhang Kevin Markham Kyle Beauchamp Kyle Kastner Lagacherie Matthieu Lars Buitinck Laurent Direr leepel Loic Esteve Luis Pedro Coelho Lukas Michel bacher maheshakya Manoj Kumar Manuel Mario Michael Krell Martin Martin Billinger Martin Ku Mateusz Susik Mathieu Blondel Matt Pico Matt Terry Matteo Visconti dOC Matti Lyra Max Linke Mehdi Cherti Michael Bommarito Michael Eickenberg Michal Romaniuk MLG mrShu Nelle Varoquaux Nicola Montecchio Nicolas Nikolay Mayorov Noel Dawe Okal Billy Olivier Grisel Óscar Nájera Paolo Puggioni Peter Prettenhofer Pratap Vardhan pvnnguyen queqichao Rafael Carrascosa Raghav R V Rahiel Kasim Randall Mason Rob Zinkov Robert Bradshaw Saket Choudhary Sam Nicholls Samuel Charron Saurabh Jha sethdandridge sinhrks snuderl Stefan Otte Stefan van der Walt Steve Tjoa swu Sylvain Zimmer tejesh95 terrycojones Thomas Delteil Thomas Unterthiner Tomas Kazmar trevorstephens tttthomasssss TzuMing Kuo ugurcaliskan ugurthemaster Vinayak Mehta Vincent Dubourg Vjacheslav Murashkin Vlad Niculae wadawson Wei Xue Will Lamond Wu Jiang x0l Xinfan Meng Yan Yi YuChin

11711 Version 0152

September 4 2014

Bug fixes

- Fixed handling of the pparameter of the Minkowski distance that was previously ignored in nearest neighbors models By Nikolay Mayorov
- Fixed duplicated alphas in linearmodelLassoLars with early stopping on 32 bit Python By Olivier Grisel and Fabian Pedregosa
- Fixed the build under Windows when scikitlearn is built with MSVC while NumPy is built with MinGW By Olivier Grisel and Federico Vaggi
- Fixed an array index overflow bug in the coordinate descent solver By Gael Varoquaux
- Better handling of numpy 19 deprecation warnings By Gael Varoquaux
- Removed unnecessary data copy in clusterKMeans By Gael Varoquaux
- Explicitly close open files to avoid ResourceWarnings under Python 3 By Calvin Giles

100 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Thetransform ofdiscriminantanalysisLinearDiscriminantAnalysis now projects the input on the most discriminant directions By Martin Billinger
- Fixed potential overflow in treesaferealloc by Lars Buitinck
- Performance optimization in isotonicIsotonicRegression By Robert Bradshaw
- nose is nonlonger a runtime dependency to import sklearn only for running the tests By Joel Nothman
- Many documentation and website fixes by Joel Nothman Lars Buitinck Matt Pico and others

11712 Version 0151

August 1 2014

Bug fixes

- Madecrossvalidationcrossvalscore usecrossvalidationKFold instead of crossvalidationStratifiedKFold on multioutput classification problems By Nikolay Mayorov
- Support unseen labels preprocessingLabelBinarizer to restore the default behavior of 0141 for backward compatibility By Hamzeh Alsalhi
- Fixed the clusterKMeans stopping criterion that prevented early convergence detection By Edward Raff and Gael Varoquaux
- Fixed the behavior of multiclassOneVsOneClassifier in case of ties at the perclass vote level by computing the correct perclass sum of prediction scores By Andreas Müller
- Madecrossvalidationcrossvalscore andgridsearchGridSearchCV accept Python lists as input data This is especially useful for crossvalidation and model selection of text processing pipelines By Andreas Müller
- Fixed data input checks of most estimators to accept input data that implements the NumPy array protocol This is the case for for pandasSeries andpandasDataFrame in recent versions of pandas By Gael Varoquaux
- Fixed a regression for linearmodelSGDClassifier withclassweightauto on data with noncontiguous labels By Olivier Grisel

11713 Version 015

July 15 2014

Highlights

- Many speed and memory improvements all across the code
- Huge speed and memory improvements to random forests and extra trees that also benefit better from parallel computing
- Incremental fit to BernoulliRBM
- AddedclusterAgglomerativeClustering for hierarchical agglomerative clustering with average linkage complete linkage and ward strategies
- AddedlinearmodelRANSACRegressor for robust regression models

117 Previous Releases 101

scikitlearn user guide Release 0213

- Added dimensionality reduction with manifoldTSNE which can be used to visualize highdimensional data

Changelog

New features

- AddedensembleBaggingClassifier andensembleBaggingRegressor metaestimators for ensembling any kind of base estimator See the Bagging section of the user guide for details and examples

By Gilles Louppe

- New unsupervised feature selection algorithm featuresselectionVarianceThreshold by Lars

Buitinck

- AddedlinearmodelRANSACRegressor metaestimator for the robust fitting of regression models By

Johannes Schönberger

- AddedclusterAgglomerativeClustering for hierarchical agglomerative clustering with average linkage complete linkage and ward strategies by Nelle Varoquaux and Gael Varoquaux
- Shorthand constructors pipelinemakepipeline andpipelinemakeunion were added by Lars

Buitinck

- Shuffle option for crossvalidationStratifiedKfold By Jeffrey Blackburne
- Incremental learning partialfit for Gaussian Naive Bayes by Imran Haque
- Addedpartialfit toBernoulliRBM By Danny Sullivan
- Addedlearningcurve utility to chart performance with respect to training size See Plotting Learning

Curves By Alexander Fabisch

- Add positive option in LassoCV andElasticNetCV By Brian Wignall and Alexandre Gramfort
- AddedlinearmodelMultiTaskElasticNetCV andlinearmodelMultiTaskLassoCV By

Manoj Kumar

- AddedmanifoldTSNE By Alexander Fabisch

Enhancements

- Add sparse input support to ensembleAdaBoostClassifier andensemble

AdaBoostRegressor metaestimators By Hamzeh Alsalhi

- Memory improvements of decision trees by Arnaud Joly
- Decision trees can now be built in bestfirst manner by using maxleafnodes as the stopping criteria

Refactored the tree code to use either a stack or a priority queue for tree building By Peter Prettenhofer and

Gilles Louppe

- Decision trees can now be fitted on fortran and cstyle arrays and noncontinuous arrays without the need to make a copy If the input array has a different dtype than npfloat32 a fortran style copy will be made since fortranstyle memory layout has speed advantages By Peter Prettenhofer and Gilles Louppe
- Speed improvement of regression trees by optimizing the the computation of the mean square error criterion

This lead to speed improvement of the tree forest and gradient boosting tree modules By Arnaud Joly

- Theimgtograph andgridtograph functions in sklearnfeatureextractionimage now returnnpndarray instead ofnpmatrix whenreturnasnpndarray See the Notes section for more information on compatibility

102 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Changed the internal storage of decision trees to use a struct array This fixed some small bugs while improving code and providing a small speed gain By Joel Nothman
 - Reduce memory usage and overhead when fitting and predicting with forests of randomized trees in parallel with `n_jobs = 1` by leveraging new threading backend of `joblib 0.8` and releasing the GIL in the tree fitting Cython code By Olivier Grisel and Gilles Louppe
 - Speed improvement of the `sklearn.ensemble.gradientboosting` module By Gilles Louppe and Peter Prettenhofer
 - Various enhancements to the `sklearn.ensemble.gradientboosting` module
 - `allow_warm_start` argument to fit additional trees
 - `max_leaf_nodes` argument to fit GBM style trees
 - `monitor` fit argument to inspect the estimator during training and refactoring of the verbose code By Peter Prettenhofer
 - `FastSklearnEnsembleExtraTrees` by caching feature values By Arnaud Joly
 - Faster depthbased tree building algorithm such as decision tree random forest extra trees or gradient tree boosting with depth based growing strategy by avoiding trying to split on found constant features in the sample subset By Arnaud Joly
 - `AdaptiveMinWeightFractionLeaf` prepruning parameter to treebased methods the minimum weighted fraction of the input samples required to be at a leaf node By Noel Dawe
 - Added `metrics.pairwise_distances_argmin_min` by Philippe Gervais
 - Added `predict` method to `cluster.AffinityPropagation` and `cluster.MeanShift` by Mathieu Blondel
 - Vector and matrix multiplications have been optimised throughout the library by Denis Engemann and Alexandre Gramfort In particular they should take less memory with older NumPy versions prior to 1.7.2
 - `PrecisionRecall` and `ROC` examples now use `train_test_split` and have more explanation of why these metrics are useful By Kyle Kastner
 - The training algorithm for `decomposition.NMF` is faster for sparse matrices and has much lower memory complexity meaning it will scale up gracefully to large datasets By Lars Buitinck
 - Added `svd_method` option with default value to “randomized” to `decomposition.FactorAnalysis` to save memory and significantly speedup computation by Denis Engemann and Alexandre Gramfort
 - Changed `cross_validation.StratifiedKFold` to try and preserve as much of the original ordering of samples as possible so as not to hide overfitting on datasets with a nonnegligible level of samples dependency By Daniel Nouri and Olivier Grisel
 - Add multioutput support to `gaussian_process.GaussianProcess` by John Novak
 - Support for precomputed distance matrices in nearest neighbor estimators by Robert Layton and Joel Nothman
 - Norm computations optimized for NumPy 1.6 and later versions by Lars Buitinck In particular the `kmeans` algorithm no longer needs a temporary data structure the size of its input
 - `DummyClassifier` can now be used to predict a constant output value By Manoj Kumar
 - `DummyRegressor` has now a `strategy` parameter which allows to predict the mean the median of the training set or a constant output value By Maheshakya Wijewardena
 - Multilabel classification output in multilabel indicator format is now supported by `metrics` `roc_auc_score` and `metrics.average_precision_score` by Arnaud Joly
 - Significant performance improvements more than 100x speedup for large problems in `isotonic` `IsotonicRegression` by Andrew Tulloch
 - Speed and memory usage improvements to the `SGD` algorithm for linear models it now uses threads not separate processes when `n_jobs=1` By Lars Buitinck
- 117 Previous Releases 103

scikitlearn user guide Release 0213

- Grid search and cross validation allow NaNs in the input arrays so that preprocessors such as preprocessingImputer can be trained within the cross validation loop avoiding potentially skewed results
- Ridge regression can now deal with sample weights in feature space only sample space until then By Michael Eickenberg Both solutions are provided by the Cholesky solver
- Several classification and regression metrics now support weighted samples with the new sampleweight argument metricsaccuracy score metricszeroone loss metricsprecision score metricsaverage precision score metrics f1 score metricsfbeta score metricsrecall score metricsrocauc score metricsexplained variance score metricsmeansquared error metrics mean absolute error metricsr2 score By Noel Dawe
- Speed up of the sample generator datasetsmake multilabel classification By Joel Nothman
- Documentation improvements
- The Working With Text Data tutorial has now been worked in to the main documentation's tutorial section Includes exercises and skeletons for tutorial presentation Original tutorial created by several authors including Olivier Grisel Lars Buitinck and many others Tutorial integration into the scikitlearn documentation by Jaques Grobler
- Added Computational Performance documentation Discussion and examples of prediction latency throughput and different factors that have influence over speed Additional tips for building faster models and choosing a relevant compromise between speed and predictive power By Eustache Diemert
- Bug fixes
- Fixed bug in decomposition MiniBatchDictionaryLearning partial fit was not working properly
- Fixed bug in linear model stochastic gradient l1 ratio was used as 10 l1 ratio
- Fixed bug in multiclass OneVsOne Classifier with string labels
- Fixed a bug in LassoCV and ElasticNetCV they would not precompute the Gram matrix with precompute True or precompute auto and n samples n features By Manoj Kumar
- Fixed incorrect estimation of the degrees of freedom in feature selection regression when variables are not centered By Virgile Fritsch
- Fixed a race condition in parallel processing with dispatch all for instance in cross_val_score By Olivier Grisel
- Raise error in cluster Feature Agglomeration and cluster Ward Agglomeration when no samples are given rather than returning meaningless clustering
- Fixed bug in gradient boosting Gradient Boosting Regressor with loss huber gamma might have not been initialized
- Fixed feature importances as computed with a forest of randomized trees when fit with sample weight None and/or with bootstrap True By Gilles Louppe

scikitlearn user guide Release 0213

API (Application Programming Interface) changes summary

- sklearnhmm is deprecated Its removal is planned for the 017 release
- Use of covarianceEllipticEnvelop has now been removed after deprecation Please use covarianceEllipticEnvelope instead
- clusterWard is deprecated Use clusterAgglomerativeClustering instead
- clusterWardClustering is deprecated Use
- clusterAgglomerativeClustering instead
- crossvalidationBootstrap is deprecated crossvalidationKFold or crossvalidationShuffleSplit are recommended instead
- Direct support for the sequence of sequences or list of lists multilabel format is deprecated To convert to and from the supported binary indicator matrix format use MultiLabelBinarizer By Joel Nothman
- Add score method to PCA (Principal Component Analysis) following the model of probabilistic PCA (Principal Component Analysis) and deprecate model whose score implementation is not correct The computation now also exploits the matrix inversion lemma for faster computation By Alexandre Gramfort
- The score method of FactorAnalysis now returns the average loglikelihood of the samples Use scoresamples to get loglikelihood of each sample By Alexandre Gramfort
- Generating boolean masks the setting indicesFalse from crossvalidation generators is deprecated Support for masks will be removed in 017 The generators have produced arrays of indices by default since 010 By Joel Nothman
- 1d arrays containing strings with dtypeobject as used in Pandas are now considered valid classification targets This fixes a regression from version 013 in some classifiers By Joel Nothman
- Fix wrong explainedvarianceratio attribute in RandomizedPCA By Alexandre Gramfort
- Fit alphas for each l1ratio instead of meanl1ratio inlinearmodelElasticNetCV and linearmodelLassoCV This changes the shape of alphas fromnalphas tonl1ratio nalphas if thel1ratio provided is a 1D array like object of length greater than one By Manoj Kumar
- FixlinearmodelElasticNetCV andlinearmodelLassoCV when fitting intercept and input data is sparse The automatic grid of alphas was not computed correctly and the scaling with normalize was wrong By Manoj Kumar
- Fix wrong maximal number of features drawn maxfeatures at each split for decision trees random forests and gradient tree boosting Previously the count for the number of drawn features started only after one non constant features in the split This bug fix will affect computational and generalization performance of those algorithms in the presence of constant features To get back previous generalization performance you should modify the value of maxfeatures By Arnaud Joly
- Fix wrong maximal number of features drawn maxfeatures at each split for ensemble ExtraTreesClassifier andensembleExtraTreesRegressor Previously only non constant features in the split was counted as drawn Now constant features are counted as drawn Furthermore at least one feature must be non constant in order to make a valid split This bug fix will affect computational and generalization performance of extra trees in the presence of constant features To get back previous generalization performance you should modify the value of maxfeatures By Arnaud Joly
- Fixutilscomputeclasseweight whenclasseweightauto Previously it was broken for input of noninteger dtype and the weighted array that was returned was wrong By Manoj Kumar
- FixcrossvalidationBootstrap to returnValueError whennttrain ntest n By Ronald Phlypo

scikitlearn user guide Release 0213

People

List of contributors for release 015 by number of commits

- 312 Olivier Grisel
- 275 Lars Buitinck
- 221 Gael Varoquaux
- 148 Arnaud Joly
- 134 Johannes Schönberger
- 119 Gilles Louppe
- 113 Joel Nothman
- 111 Alexandre Gramfort
- 95 Jaques Grobler
- 89 Denis Engemann
- 83 Peter Prettenhofer
- 83 Alexander Fabisch
- 62 Mathieu Blondel
- 60 Eustache Diemert
- 60 Nelle Varoquaux
- 49 Michael Bommarito
- 45 ManojKumarS
- 28 Kyle Kastner
- 26 Andreas Mueller
- 22 Noel Dawe
- 21 Maheshakya Wijewardena
- 21 Brooke Osborn
- 21 Hamzeh Alsalhi
- 21 Jake VanderPlas
- 21 Philippe Gervais
- 19 Bala Subrahmanyam Varanasi
- 12 Ronald Phlypo
- 10 Mikhail Korobov
- 8 Thomas Unterthiner
- 8 Jeffrey Blackburne
- 8 eltermann
- 8 bwignall
- 7 Ankit Agrawal
- 7 CJ Carey

scikitlearn user guide Release 0213

- 6 Daniel Nouri
- 6 Chen Liu
- 6 Michael Eickenberg
- 6 ugurthemaster
- 5 Aaron Schumacher
- 5 Baptiste Lagarde
- 5 Rajat Khanduja
- 5 Robert McGibbon
- 5 Sergio Pascual
- 4 Alexis Metaireau
- 4 Ignacio Rossi
- 4 Virgile Fritsch
- 4 Sebastian Säger
- 4 Ilambharathi Kanniah
- 4 sdenton4
- 4 Robert Layton
- 4 Alyssa
- 4 Amos Waterland
- 3 Andrew Tulloch
- 3 murad
- 3 Steven Maude
- 3 Karol Pysniak
- 3 Jacques Kvam
- 3 cgohlke
- 3 cjlin
- 3 Michael Becker
- 3 hamzeh
- 3 Eric Jacobsen
- 3 john collins
- 3 kaushik94
- 3 Erwin Marsi
- 2 csytracy
- 2 LK
- 2 Vlad Niculae
- 2 Laurent Direr
- 2 Erik Shilts

scikitlearn user guide Release 0213

- 2 Raul Garreta
 - 2 Yoshiki Vázquez Baeza
 - 2 Yung Siang Liao
 - 2 abhishek thakur
 - 2 James Yu
 - 2 Rohit Sivaprasad
 - 2 Roland Szabo
 - 2 amormachine
 - 2 Alexis Mignon
 - 2 Oscar Carlsson
 - 2 Nantas Nardelli
 - 2 jess010
 - 2 kowalski87
 - 2 Andrew Clegg
 - 2 Federico Vaggi
 - 2 Simon Frid
 - 2 FélixAntoine Fortin
 - 1 Ralf Gommers
 - 1 taft
 - 1 Ronan Amicel
 - 1 Rupesh Kumar Srivastava
 - 1 Ryan Wang
 - 1 Samuel Charron
 - 1 Samuel StJean
 - 1 Fabian Pedregosa
 - 1 Skipper Seabold
 - 1 Stefan Walk
 - 1 Stefan van der Walt
 - 1 Stephan Hoyer
 - 1 Allen Riddell
 - 1 Valentin Haenel
 - 1 Vijay Ramesh
 - 1 Will Myers
 - 1 Yaroslav Halchenko
 - 1 Yoni BenMeshulam
 - 1 Yury V Zaytsev
- 108 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- 1 adrinjalali
- 1 ai8rahim
- 1 alemagnani
- 1 alex
- 1 benjamin wilson
- 1 chalmerlowe
- 1 dzikie dro `zd`ze
- 1 jamestwebber
- 1 matrixorz
- 1 popo
- 1 samuela
- 1 François Boulogne
- 1 Alexander Measure
- 1 Ethan White
- 1 Guilherme Trein
- 1 Hendrik Heuer
- 1 IvicaJovic
- 1 Jan Hendrik Metzen
- 1 Jean Michel Rouly
- 1 Eduardo Ariño de la Rubia
- 1 Jelle Zijlstra
- 1 Eddy L O Jansson
- 1 Denis
- 1 John
- 1 John Schmidt
- 1 Jorge Cañardo Alastuey
- 1 Joseph Perla
- 1 Joshua Vredevoogd
- 1 José Ricardo
- 1 Julien Miotte
- 1 Kemal Eren
- 1 Kenta Sato
- 1 David Cournapeau
- 1 Kyle Kelley
- 1 Daniele Medri
- 1 Laurent Luce

scikitlearn user guide Release 0213

- 1 Laurent Pierron
- 1 Luis Pedro Coelho
- 1 DanielWeitzenfeld
- 1 Craig Thompson
- 1 ChyiKwei Yau
- 1 Matthew Brett
- 1 Matthias Feurer
- 1 Max Linke
- 1 Chris Filo Gorgolewski
- 1 Charles Earl
- 1 Michael Hanke
- 1 Michele Orrù
- 1 Bryan Lunt
- 1 Brian Kearns
- 1 Paul Butler
- 1 Paweł Mandra
- 1 Peter
- 1 Andrew Ash
- 1 Pietro Zambelli
- 1 staubda

11714 Version 014

August 7 2013

Changelog

- Missing values with sparse and dense matrices can be imputed with the transformer preprocessing Imputer by Nicolas Trésegnie
- The core implementation of decisions trees has been rewritten from scratch allowing for faster tree induction and lower memory consumption in all treebased estimators By Gilles Louppe
- AddedensembleAdaBoostClassifier andensembleAdaBoostRegressor by Noel Dawe and Gilles Louppe See the AdaBoost section of the user guide for details and examples
- AddedgridsearchRandomizedSearchCV andgridsearchParameterSampler for randomized hyperparameter optimization By Andreas Müller
- Added biclustering algorithms sklearnclusterbiclusterSpectralCoclustering and sklearnclusterbiclusterSpectralBiclustering data generation methods sklearn datasetsmakebiclusters andsklearndatasetsmakecheckerboard and scoring metrics sklearnmetricsconsensussscore By Kemal Eren
- Added Restricted Boltzmann Machines neuralnetworkBernoulliRBM By Yann Dauphin

110 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Python 3 support by Justin Vincent Lars Buitinck Subhdeep Moitra and Olivier Grisel All tests now pass under Python 33
 - Ability to pass one penalty alpha value per target in linearmodelRidge by eickenberg and Mathieu Blondel
 - Fixedsklearnlinearmodelstochasticgradientpy L2 regularization issue minor practical significance By Norbert Crombach and Mathieu Blondel
 - Added an interactive version of Andreas Müller’s Machine Learning Cheat Sheet for scikitlearn to the documentation See Choosing the right estimator By Jaques Grobler
 - gridsearchGridSearchCV andcrossvalidationcrossvalscore now support the use of advanced scoring function such as area under the ROC curve and fbeta scores See The scoring parameter defining model evaluation rules for details By Andreas Müller and Lars Buitinck Passing a function from sklearnmetrics asscorefunc is deprecated
 - Multilabel classification output is now supported by metricsaccuracy score metricszeroone loss metricsf1 score metricsfbeta score metrics classificationreport metricsprecision score andmetricsrecall score by Arnaud Joly
 - Two new metrics metricshammingloss andmetricsjaccardsimilarity score are added with multilabel support by Arnaud Joly
 - Speed and memory usage improvements in featureextractiontextCountVectorizer and featureextractiontextTfidfVectorizer by Jochen Wersdörfer and Roman Sinayev
 - Themindf parameter in featureextractiontextCountVectorizer and featureextractiontextTfidfVectorizer which used to be 2 has been reset to 1 to avoid unpleasant surprises empty vocabularies for novice users who try it out on tiny document collections A value of at least 2 is still recommended for practical use
 - svmLinearSVC linearmodelSGDClassifier andlinearmodelSGDRegressor now have asparsify method that converts their coef into a sparse matrix meaning stored models trained using these estimators can be made much more compact
 - linearmodelSGDClassifier now produces multiclass probability estimates when trained under log loss or modified Huber loss
 - Hyperlinks to documentation in example code on the website by Martin Luessi
 - Fixed bug in preprocessingMinMaxScaler causing incorrect scaling of the features for nondefault featurerange settings By Andreas Müller
 - maxfeatures intreeDecisionTreeClassifier treeDecisionTreeRegressor and all derived ensemble estimators now supports percentage values By Gilles Louppe
 - Performance improvements in isotonicIsotonicRegression by Nelle Varoquaux
 - metricsaccuracy score has an option normalize to return the fraction or the number of correctly classified sample by Arnaud Joly
 - Addedmetricslogloss that computes log loss aka crossentropy loss By Jochen Wersdörfer and Lars Buitinck
 - A bug that caused ensembleAdaBoostClassifier ’s to output incorrect probabilities has been fixed
 - Feature selectors now share a mixin providing consistent transform inversetransform and getsupport methods By Joel Nothman
 - A fittedgridsearchGridSearchCV orgridsearchRandomizedSearchCV can now generally be pickled By Joel Nothman
- 117 Previous Releases 111

scikitlearn user guide Release 0213

- Refactored and vectorized implementation of metricsroccurve andmetrics precisionrecallcurve By Joel Nothman
- The new estimator sklearncompositionTruncatedSVD performs dimensionality reduction using SVD on sparse matrices and can be used for latent semantic analysis LSA By Lars Buitinck
- Added selfcontained example of outofcore learning on text data Outofcore classification of text documents By Eustache Diemert
- The default number of components for sklearncompositionRandomizedPCA is now correctly documented to be nfeatures This was the default behavior so programs using it will continue to work as they did
- sklearnclusterKMeans now fits several orders of magnitude faster on sparse data the speedup depends on the sparsity By Lars Buitinck
- Reduce memory footprint of FastICA by Denis Engemann and Alexandre Gramfort
- Verbose output in sklearnensemblegradientboosting now uses a column format and prints progress in decreasing frequency It also shows the remaining time By Peter Prettenhofer
- sklearnensemblegradientboosting provides outofbag improvement oobimprovement rather than the OOB score for model selection An example that shows how to use OOB estimates to select the number of trees was added By Peter Prettenhofer
- Most metrics now support string labels for multiclass classification by Arnaud Joly and Lars Buitinck
- New OrthogonalMatchingPursuitCV class by Alexandre Gramfort and Vlad Niculae
- Fixed a bug in sklearncovarianceGraphLassoCV the ‘alphas’ parameter now works as expected when given a list of values By Philippe Gervais
- Fixed an important bug in sklearncovarianceGraphLassoCV that prevented all folds provided by a CV (Cross-Validation) object to be used only the first 3 were used When providing a CV (Cross-Validation) object execution time may thus increase significantly compared to the previous version bug results are correct now By Philippe Gervais
- crossvalidationcrossvalscore and thegridsearch module is now tested with multi output data by Arnaud Joly
- datasetsmakemultilabelclassification can now return the output in label indicator multilabel format by Arnaud Joly
- Knearest neighbors neighborsKNeighborsRegressor andneighbors RadiusNeighborsRegressor and radius neighbors neighborsRadiusNeighborsRegressor andneighborsRadiusNeighborsClassifier support multioutput data by Arnaud Joly
- Random state in LibSVMbased estimators svmSVC NuSVC OneClassSVM svmSVR svmNuSVR can now be controlled This is useful to ensure consistency in the probability estimates for the classifiers trained withprobabilityTrue By Vlad Niculae
- Outofcore learning support for discrete naive Bayes classifiers sklearnnaivebayes MultinomialNB andsklearnnaivebayesBernoulliNB by adding the partialfit method by Olivier Grisel
- New website design and navigation by Gilles Louppe Nelle Varoquaux Vincent Michel and Andreas Müller
- Improved documentation on multiclass multilabel and multioutput classification by Yannick Schwartz and Arnaud Joly
- Better input and error handling in the metrics module by Arnaud Joly and Joel Nothman
- Speed optimization of the hmm module by Mikhail Korobov
- Significant speed improvements for sklearnclusterDBSCAN by cleverless

scikitlearn user guide Release 0213

API (Application Programming Interface) changes summary

- Theaucscore was renamed rocaucscore
 - Testing scikitlearn with sklearntest is deprecated Use nosetests sklearn from the command line
 - Feature importances in treeDecisionTreeClassifier treeDecisionTreeRegressor and all derived ensemble estimators are now computed on the fly when accessing the featureimportances attribute Setting computeimportancesTrue is no longer required By Gilles Louppe
 - linearmodellassopath andlinearmodelenetpath can return its results in the same format as that oflinearmodelarspath This is done by setting the returnmodels parameter to False By Jaques Grobler and Alexandre Gramfort
 - gridsearchIterGrid was renamed to gridsearchParameterGrid
 - Fixed bug in KFold causing imperfect class balance in some cases By Alexandre Gramfort and Tadej Janež
 - sklearnneighborsBallTree has been refactored and a sklearnneighborsKDTree has been added which shares the same interface The Ball Tree now works with a wide variety of distance metrics Both classes have many new methods including singletree and dualtree queries breadthfirst and depthfirst searching and more advanced queries such as kernel density estimation and 2point correlation functions By Jake Vanderplas
 - Support for scipyspatialcKDTree within neighbors queries has been removed and the functionality replaced with the new KDTree class
 - sklearnneighborsKernelDensity has been added which performs efficient kernel density estimation with a variety of kernels
 - sklearndecompositionKernelPCA now always returns output with ncomponents components unless the new parameter removezeroeig is set toTrue This new behavior is consistent with the way kernel PCA (Principal Component Analysis) was always documented previously the removal of components with zero eigenvalues was tacitly performed on all data
 - gcvmodeauto no longer tries to perform SVD on a densified sparse matrix in sklearn
 - linearmodelRidgeCV
 - Sparse matrix support in sklearndecompositionRandomizedPCA is now deprecated in favor of the newTruncatedSVD
 - crossvalidationKFold andcrossvalidationStratifiedKFold now enforce nolds 2 otherwise a ValueError is raised By Olivier Grisel
 - datasetsloadfiles 'scharset andcharseterrors parameters were renamed encoding and decodeerrors
 - Attribute oobscore in sklearnensembleGradientBoostingRegressor and sklearnensembleGradientBoostingClassifier is deprecated and has been replaced by oobimprovement
 - Attributes in OrthogonalMatchingPursuit have been deprecated copyX Gram and precompute_gram renamed precompute for consistency See 2224
 - sklearnpreprocessingStandardScaler now converts integer input to float and raises a warning Previously it rounded for dense integer input
 - sklearnmulticlassOneVsRestClassifier now has a decisionfunction method This will return the distance of each sample from the decision boundary for each class as long as the underlying estimators implement the decisionfunction method By Kyle Kastner
 - Better input validation warning on unexpected shapes for y
- 117 Previous Releases 113

scikitlearn user guide Release 0213

People

List of contributors for release 014 by number of commits

- 277 Gilles Louppe
- 245 Lars Buitinck
- 187 Andreas Mueller
- 124 Arnaud Joly
- 112 Jaques Grobler
- 109 Gael Varoquaux
- 107 Olivier Grisel
- 102 Noel Dawe
- 99 Kemal Eren
- 79 Joel Nothman
- 75 Jake VanderPlas
- 73 Nelle Varoquaux
- 71 Vlad Niculae
- 65 Peter Prettenhofer
- 64 Alexandre Gramfort
- 54 Mathieu Blondel
- 38 Nicolas Trésegne
- 35 eustache
- 27 Denis Engemann
- 25 Yann N Dauphin
- 19 Justin Vincent
- 17 Robert Layton
- 15 Doug Coleman
- 14 Michael Eickenberg
- 13 Robert Marchman
- 11 Fabian Pedregosa
- 11 Philippe Gervais
- 10 Jim Holmström
- 10 Tadej Janež
- 10 syhw
- 9 Mikhail Korobov
- 9 Steven De Gryze
- 8 sergeyf
- 7 Ben Root

114 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- 7 Hrishikesh Huilgolkar
- 6 Kyle Kastner
- 6 Martin Luessi
- 6 Rob Speer
- 5 Federico Vaggi
- 5 Raul Garreta
- 5 Rob Zinkov
- 4 Ken Geis
- 3 A Flaxman
- 3 Denton Cockburn
- 3 Dougal Sutherland
- 3 Ian Ozsvald
- 3 Johannes Schönberger
- 3 Robert McGibbon
- 3 Roman Sinayev
- 3 Szabo Roland
- 2 Diego Molla
- 2 Imran Haque
- 2 Jochen Wersdörfer
- 2 Sergey Karayev
- 2 Yannick Schwartz
- 2 jamestwebber
- 1 Abhijeet Kolhe
- 1 Alexander Fabisch
- 1 Bastiaan van den Berg
- 1 Benjamin Peterson
- 1 Daniel Velkov
- 1 Fazlul Shahriar
- 1 Felix Brockherde
- 1 FélixAntoine Fortin
- 1 Harikrishnan S
- 1 Jack Hale
- 1 JakeMick
- 1 James McDermott
- 1 John Benediktsson
- 1 John Zwinck

scikitlearn user guide Release 0213

- 1 Joshua Vredevoogd
- 1 Justin Pati
- 1 Kevin Hughes
- 1 Kyle Kelley
- 1 Matthias Ekman
- 1 Miroslav Shubernetskiy
- 1 Naoki Orii
- 1 Norbert Crombach
- 1 Rafael Cunha de Almeida
- 1 Rolando Espinoza La fuente
- 1 Seamus Abshire
- 1 Sergey Feldman
- 1 Sergio Medina
- 1 Stefano Lattarini
- 1 Steve Koch
- 1 Sturla Molden
- 1 Thomas Jarosch
- 1 Yaroslav Halchenko

11715 Version 0131

February 23 2013

The 0131 release only fixes some bugs and does not add any new functionality

Changelog

- Fixed a testing error caused by the function crossvalidationtraintestsplit being interpreted as a test by Yaroslav Halchenko
- Fixed a bug in the reassignment of small clusters in the clusterMiniBatchKMeans by Gael Varoquaux
- Fixed default value of gamma indecompositionKernelPCA by Lars Buitinck
- Updated joblib to 070d by Gael Varoquaux
- Fixed scaling of the deviance in ensembleGradientBoostingClassifier by Peter Prettenhofer
- Better tiebreaking in multiclassOneVsOneClassifier by Andreas Müller
- Other small improvements to tests and documentation

116 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

People

List of contributors for release 0131 by number of commits

- 16 Lars Buitinck
- 12 Andreas Müller
- 8 Gael Varoquaux
- 5 Robert Marchman
- 3 Peter Prettenhofer
- 2 Hrishikesh Huilgolkar
- 1 Bastiaan van den Berg
- 1 Diego Molla
- 1 Gilles Louppe
- 1 Mathieu Blondel
- 1 Nelle Varoquaux
- 1 Rafael Cunha de Almeida
- 1 Rolando Espinoza La fuente
- 1 Vlad Niculae
- 1 Yaroslav Halchenko

11716 Version 013

January 21 2013

New Estimator Classes

- dummyDummyClassifier anddummyDummyRegressor two dataindependent predictors by Mathieu Blondel Useful to sanitycheck your estimators See Dummy estimators in the user guide Multioutput support added by Arnaud Joly
- decompositionFactorAnalysis a transformer implementing the classical factor analysis by Chris tian Osendorfer and Alexandre Gramfort See Factor Analysis in the user guide
- featureextractionFeatureHasher a transformer implementing the “hashing trick” for fast lowmemory feature extraction from string fields by Lars Buitinck and featureextractiontext HashingVectorizer for text documents by Olivier Grisel See Feature hashing andVectorizing a large text corpus with the hashing trick for the documentation and sample usage
- pipelineFeatureUnion a transformer that concatenates results of several other transformers by Andreas Müller See FeatureUnion composite feature spaces in the user guide
- randomprojectionGaussianRandomProjection randomprojection SparseRandomProjection and the function randomprojection
- johnsonlindenstraussmindim The first two are transformers implementing Gaussian and sparse random projection matrix by Olivier Grisel and Arnaud Joly See Random Projection in the user guide
- kernelapproximationNystroem a transformer for approximating arbitrary kernels by Andreas Müller See Nystroem Method for Kernel Approximation in the user guide

117 Previous Releases 117

scikitlearn user guide Release 0213

- preprocessingOneHotEncoder a transformer that computes binary encodings of categorical features by Andreas Müller See Encoding categorical features in the user guide
 - linearmodelPassiveAggressiveClassifier and linearmodelPassiveAggressiveRegressor predictors implementing an efficient stochastic optimization for linear models by Rob Zinkov and Mathieu Blondel See Passive Aggressive Algorithms in the user guide
 - ensembleRandomTreesEmbedding a transformer for creating highdimensional sparse representations using ensembles of totally random trees by Andreas Müller See Totally Random Trees Embedding in the user guide
 - manifoldSpectralEmbedding and function manifoldspectralembedding implementing the “laplacian eigenmaps” transformation for nonlinear dimensionality reduction by Wei Li See Spectral Embedding in the user guide
 - isotonicIsotonicRegression by Fabian Pedregosa Alexandre Gramfort and Nelle Varoquaux
- Changelog
- metricszeroone loss formerly metricszeroone now has option for normalized output that reports the fraction of misclassifications rather than the raw number of misclassifications By Kyle Beauchamp
 - treeDecisionTreeClassifier and all derived ensemble models now support sample weighting by Noel Dawe and Gilles Louppe
 - Speedup improvement when using bootstrap samples in forests of randomized trees by Peter Prettenhofer and Gilles Louppe
 - Partial dependence plots for Gradient Tree Boosting in ensemble partialdependencepartialdependence by Peter Prettenhofer See sphxglrautoexamplesensembleplotpartialdependencepy for an example
 - The table of contents on the website has now been made expandable by Jaques Grobler
 - featureselectionSelectPercentile now breaks ties deterministically instead of returning all equally ranked features
 - featureselectionSelectKBest and featureselectionSelectPercentile are more numerically stable since they use scores rather than pvalues to rank results This means that they might sometimes select different features than they did previously
 - Ridge regression and ridge classification fitting with sparsecg solver no longer has quadratic memory complexity by Lars Buitinck and Fabian Pedregosa
 - Ridge regression and ridge classification now support a new fast solver called lsqr by Mathieu Blondel
 - Speed up of metricsprecisionrecallcurve by Conrad Lee
 - Added support for reading/writing svmlight files with pairwise preference attribute qid in svmlight file format in datasetsdumpsvmlightfile and datasetsloadsvmlightfile by Fabian Pedregosa
 - Faster and more robust metricsconfusionmatrix and Clustering performance evaluation by Wei Li
 - crossvalidationcrossvalscore now works with precomputed kernels and affinity matrices by Andreas Müller
 - LARS algorithm made more numerically stable with heuristics to drop regressors too correlated as well as to stop the path when numerical noise becomes predominant by Gael Varoquaux
 - Faster implementation of metricsprecisionrecallcurve by Conrad Lee
 - New kernel metricschi2kernel by Andreas Müller often used in computer vision applications

scikitlearn user guide Release 0213

- Fix of longstanding bug in naivebayesBernoulliNB fixed by Shaun Jackman
 - Implemented predict_proba in multiclassOneVsRestClassifier by Andrew Winterman
 - Improve consistency in gradient boosting estimators ensembleGradientBoostingRegressor and ensembleGradientBoostingClassifier use the estimator treeDecisionTreeRegressor instead of the treeTree data structure by Arnaud Joly
 - Fixed a floating point exception in the decision trees module by Seberg
 - Fix metrics roc_curve fails when y_true has only one class by Wei Li
 - Add the metrics mean_absolute_error function which computes the mean absolute error The metrics mean_squared_error metrics mean_absolute_error and metrics r2_score metrics support multioutput by Arnaud Joly
 - Fixed class_weight support in svmLinearSVC and linear_model LogisticRegression by Andreas Müller The meaning of class_weight was reversed as erroneously higher weight meant less positives of a given class in earlier releases
 - Improve narrative documentation and consistency in sklearn.metrics for regression and classification metrics by Arnaud Joly
 - Fixed a bug in sklearn.svm.SVC when using csrmatrices with unsorted indices by Xinfan Meng and Andreas Müller
 - MiniBatchKMeans Add random reassignment of cluster centers with little observations attached to them by Gael Varoquaux
- API (Application Programming Interface) changes summary
- Renamed all occurrences of n_atoms to n_components for consistency This applies to decomposition DictionaryLearning decomposition MiniBatchDictionaryLearning decomposition dict_learning decomposition dict_learning_online
 - Renamed all occurrences of max_iter to max_iter for consistency This applies to semisupervised LabelPropagation and semisupervised label_propagation LabelSpreading
 - Renamed all occurrences of learn_rate to learning_rate for consistency in ensemble BaseGradientBoosting and ensemble GradientBoostingRegressor
 - The module sklearn.linear_model.sparse is gone Sparse matrix support was already integrated into the “regular” linear models
 - sklearn.metrics.mean_squared_error which incorrectly returned the accumulated error was removed Use mean_squared_error instead
 - Passing class_weight parameters to fit methods is no longer supported Pass them to estimator constructors instead
 - GMMs no longer have decode_and_rvs methods Use the score_predict_or_sample methods instead
 - The solver_fit option in Ridge regression and classification is now deprecated and will be removed in v0.14 Use the constructor option instead
 - feature_extraction.text.DictVectorizer now returns sparse matrices in the CSR (Compressed Sparse Row) format instead of COO
 - Renamed kinkcrossvalidationKFold and crossvalidationStratifiedKFold to kfold renamed n_bootstraps to n_iter in crossvalidation Bootstrap
- 117 Previous Releases 119

scikitlearn user guide Release 0.21.3

- Renamed all occurrences of iterations toniter for consistency This applies to crossvalidationShuffleSplit crossvalidationStratifiedShuffleSplit
utilsrandomizedrangefinder andutilsrandomizedsvd
- Replaced rho inlinearmodelElasticNet andlinearmodelSGDClassifier by l1ratio The rho parameter had different meanings l1ratio was introduced to avoid confusion It has the same meaning as previously rho inlinearmodelElasticNet andl1rho in linearmodelSGDClassifier
- linearmodelLassoLars andlinearmodelLars now store a list of paths in the case of multiple targets rather than an array of paths
- The attribute gmm ofhmmGMMHMM was renamed to gmm to adhere more strictly with the API (Application Programming Interface)
- clusterspectralembedding was moved to manifoldspectralembedding
- Renamed eigtol inmanifoldspectralembedding clusterSpectralClustering to eigentol renamedmode toeigensolver
- Renamed mode inmanifoldspectralembedding andclusterSpectralClustering to eigensolver
- classes andn_classes attributes of treeDecisionTreeClassifier and all derived ensemble models are now flat in case of single output problems and nested in case of multioutput problems
- Theestimators attribute of ensemblegradientboostingGradientBoostingRegressor andensemblegradientboostingGradientBoostingClassifier is now an array of class'treeDecisionTreeRegressor'
- Renamed chunksize tobatchsize indecompositionMiniBatchDictionaryLearning anddecompositionMiniBatchSparsePCA for consistency
- svmSVC andsvmNuSVC now provide a classes attribute and support arbitrary dtypes for labels y Also the dtype returned by predict now reflects the dtype of yduringfit used to be npfloat
- Changed default testsize in crossvalidationtraintestsplit to None added possibility to infer testsize fromtrainsize incrossvalidationShuffleSplit and crossvalidationStratifiedShuffleSplit
- Renamed function sklearnmetricszeroone tosklearnmetricszeroone_loss Be aware that the default behavior in sklearnmetricszeroone_loss is different from sklearn metricszeroone normalizeFalse is changed to normalizeTrue
- Renamed function metricszeroonescore tometricsaccuracy_score
- datasetsmake_circles now has the same number of inner and outer points
- In the Naive Bayes classifiers the classprior parameter was moved from fit toinit

People

List of contributors for release 0.13 by number of commits

- 364 Andreas Müller
- 143 Arnaud Joly
- 137 Peter Prettenhofer
- 131 Gael Varoquaux
- 117 Mathieu Blondel
- 108 Lars Buitinck

120 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- 106 Wei Li
- 101 Olivier Grisel
- 65 Vlad Niculae
- 54 Gilles Louppe
- 40 Jaques Grobler
- 38 Alexandre Gramfort
- 30 Rob Zinkov
- 19 Aymeric Masurelle
- 18 Andrew Winterman
- 17 Fabian Pedregosa
- 17 Nelle Varoquaux
- 16 Christian Osendorfer
- 14 Daniel Nouri
- 13 Virgile Fritsch
- 13 syhw
- 12 Satrajit Ghosh
- 10 Corey Lynch
- 10 Kyle Beauchamp
- 9 Brian Cheung
- 9 Immanuel Bayer
- 9 mrShu
- 8 Conrad Lee
- 8 James Bergstra
- 7 Tadej Janež
- 6 Brian Cajes
- 6 Jake Vanderplas
- 6 Michael
- 6 Noel Dawe
- 6 Tiago Nunes
- 6 cow
- 5 Anze
- 5 Shiqiao Du
- 4 Christian Jauvin
- 4 Jacques Kvam
- 4 Richard T Guy
- 4 Robert Layton

scikitlearn user guide Release 0213

- 3 Alexandre Abraham
- 3 Doug Coleman
- 3 Scott Dickerson
- 2 ApproximateIdentity
- 2 John Benediktsson
- 2 Mark Veronda
- 2 Matti Lyra
- 2 Mikhail Korobov
- 2 Xinfan Meng
- 1 Alejandro Weinstein
- 1 Alexandre Passos
- 1 Christoph Deil
- 1 Eugene Nizhibitsky
- 1 Kenneth C Arnold
- 1 Luis Pedro Coelho
- 1 Miroslav Batchkarov
- 1 Pavel
- 1 Sebastian Berg
- 1 Shaun Jackman
- 1 Subhodeep Moitra
- 1 bob
- 1 dengemann
- 1 emanuele
- 1 x006

11717 Version 0121

October 8 2012

The 0121 release is a bugfix release with no additional features but is instead a set of bug fixes

Changelog

- Improved numerical stability in spectral embedding by Gael Varoquaux
- Doctest under windows 64bit by Gael Varoquaux
- Documentation fixes for elastic net by Andreas Müller and Alexandre Gramfort
- Proper behavior with fortranordered NumPy arrays by Gael Varoquaux
- Make GridSearchCV work with nonCSR sparse matrix by Lars Buitinck
- Fix parallel computing in MDS by Gael Varoquaux

122 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Fix Unicode support in count vectorizer by Andreas Müller
- Fix MinCovDet breaking with Xshape 3 1 by Virgile Fritsch
- Fix clone of SGD objects by Peter Prettenhofer
- Stabilize GMM by Virgile Fritsch

People

- 14 Peter Prettenhofer
- 12 Gael Varoquaux
- 10 Andreas Müller
- 5 Lars Buitinck
- 3 Virgile Fritsch
- 1 Alexandre Gramfort
- 1 Gilles Louppe
- 1 Mathieu Blondel

11718 Version 012

September 4 2012

Changelog

- Various speed improvements of the decision trees module by Gilles Louppe
- ensembleGradientBoostingRegressor and ensembleGradientBoostingClassifier now support feature subsampling via the maxfeatures argument by Peter Prettenhofer
- Added Huber and Quantile loss functions to ensembleGradientBoostingRegressor by Peter Prettenhofer
- Decision trees and forests of randomized trees now support multioutput classification and regression problems by Gilles Louppe
- Added preprocessingLabelEncoder a simple utility class to normalize labels or transform non numerical labels by Mathieu Blondel
- Added the epsilon insensitive loss and the ability to make probabilistic predictions with the modified huber loss in Stochastic Gradient Descent by Mathieu Blondel
- Added Multidimensional Scaling MDS by Nelle Varoquaux
- SVMlight file format loader now detects compressed gzip/bzip2 files and decompresses them on the fly by Lars Buitinck
- SVMlight file format serializer now preserves double precision floating point values by Olivier Grisel
- A common testing framework for all estimators was added by Andreas Müller
- Understandable error messages for estimators that do not accept sparse input by Gael Varoquaux
- Speedups in hierarchical clustering by Gael Varoquaux In particular building the tree now supports early stopping This is useful when the number of clusters is not small compared to the number of samples

117 Previous Releases 123

scikitlearn user guide Release 0213

- Add MultiTaskLasso and MultiTaskElasticNet for joint feature selection by Alexandre Gramfort
- Added metrics auc score and metrics average precision score convenience functions by Andreas Müller
- Improved sparse matrix support in the Feature selection module by Andreas Müller
- New word boundaries aware character ngram analyzer for the Text feature extraction module by kernc
- Fixed bug in spectral clustering that led to single point clusters by Andreas Müller
- In feature extraction text CountVectorizer added an option to ignore infrequent words

mindf by Andreas Müller

- Add support for multiple targets in some linear models ElasticNet Lasso and Orthogonal Matching Pursuit by Vlad Niculae and Alexandre Gramfort
- Fixes indecomposition Probabilistic PCA score function by Wei Li
- Fixed feature importance computation in Gradient Tree Boosting

API (Application Programming Interface) changes summary

- The old scikitslearn package has disappeared all code should import from sklearn instead which was introduced in 09
- In metrics roc curve the thresholds array is now returned with it's order reversed in order to keep it consistent with the order of the returned fpr and tpr
- In hmm objects like hmm GaussianHMM hmm MultinomialHMM etc all parameters must be passed to the object when initialising it and not through fit Now fit will only accept the data as an input parameter
- For all SVM (Support Vector Machine) classes a faulty behavior of gamma was fixed Previously the default gamma value was only computed the first time fit was called and then stored It is now recalculated on every call to fit
- All Base classes are now abstract meta classes so that they can not be instantiated
- cluster WardTree now also returns the parent array This is necessary for early stopping in which case the tree is not completely built
- In feature extraction text CountVectorizer the parameters minn and maxn were joined to the parameter ngram range to enable grid searching both at once
- In feature extraction text CountVectorizer words that appear only in one document are now ignored by default To reproduce the previous behavior set mindf1
- Fixed API (Application Programming Interface) inconsistency linear model SGD Classifier predict_proba now returns 2d array when fit on two classes
- Fixed API (Application Programming Interface) inconsistency discriminant analysis Quadratic Discriminant Analysis decision function and discriminant analysis Linear Discriminant Analysis decision function now return 1d arrays when fit on two classes
- Grid of alphas used for fitting linear model Lasso CV and linear model Elastic Net CV is now stored in the attribute alphas rather than overriding the init parameter alphas
- Linear models when alpha is estimated by cross validation store the estimated value in the alpha attribute rather than just alpha or best alpha
- ensemble Gradient Boosting Classifier now supports ensemble Gradient Boosting Classifier staged predict_proba and ensemble Gradient Boosting Classifier staged predict

scikitlearn user guide Release 0213

•svmparsesVC and other sparse SVM (Support Vector Machine) classes are now deprecated The all classes in the Support Vector Machines module now automatically select the sparse or dense representation base on the input

• All clustering algorithms now interpret the array Xgiven tofit as input data in particular cluster

SpectralClustering andclusterAffinityPropagation which previously expected affinity matrices

• For clustering algorithms that take the desired number of clusters as a parameter this parameter is now called nclusters

People

- 267 Andreas Müller
- 94 Gilles Louppe
- 89 Gael Varoquaux
- 79 Peter Prettenhofer
- 60 Mathieu Blondel
- 57 Alexandre Gramfort
- 52 Vlad Niculae
- 45 Lars Buitinck
- 44 Nelle Varoquaux
- 37 Jaques Grobler
- 30 Alexis Mignon
- 30 Immanuel Bayer
- 27 Olivier Grisel
- 16 Subhdeep Moitra
- 13 Yannick Schwartz
- 12 kernc
- 11 Virgile Fritsch
- 9 Daniel Duckworth
- 9 Fabian Pedregosa
- 9 Robert Layton
- 8 John Benediktsson
- 7 Marko Burjek
- 5 Nicolas Pinto
- 4 Alexandre Abraham
- 4 Jake Vanderplas
- 3 Brian Holt
- 3 Edouard Duchesnay
- 3 Florian Hoenig

scikitlearn user guide Release 0213

- 3 flyingimidev
- 2 Francois Savard
- 2 Hannes Schulz
- 2 Peter Welinder
- 2 Yaroslav Halchenko
- 2 Wei Li
- 1 Alex Companioni
- 1 Brandyn A White
- 1 Bussonnier Matthias
- 1 CharlesPierre Astolfi
- 1 Dan O’Huiginn
- 1 David Cournapeau
- 1 Keith Goodman
- 1 Ludwig Schwardt
- 1 Olivier Hervieu
- 1 Sergio Medina
- 1 Shiqiao Du
- 1 Tim SheermanChase
- 1 buguen

11719 Version 011

May 7 2012

Changelog

Highlights

- Gradient boosted regression trees Gradient Tree Boosting for classification and regression by Peter Prettenhofer and Scott White
- Simple dictbased feature loader with support for categorical variables featureextraction DictVectorizer by Lars Buitinck
- Added Matthews correlation coefficient metricsmatthewscore and added macro and micro average options to metricsprecisionscore metricsrecallscore andmetricsf1score by Satrajit Ghosh
- Out of Bag Estimates of generalization error for Ensemble methods by Andreas Müller
- Randomized sparse linear models for feature selection by Alexandre Gramfort and Gael Varoquaux
- Label Propagation for semisupervised learning by Clay Woolam Note the semisupervised API (Application Programming Interface) is still work in progress and may change

126 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Added BICAIC model selection to classical Gaussian mixture models and unified the API (Application Programming Interface) with the remainder of scikitlearn by Bertrand Thirion
- AddedsklearncrossvalidationStratifiedShuffleSplit which is a sklearn crossvalidationShuffleSplit with balanced splits by Yannick Schwartz
- sklearnneighborsNearestCentroid classifier added along with a shrinkthreshold parameter which implements shrunken centroid classification by Robert Layton

Other changes

- Merged dense and sparse implementations of Stochastic Gradient Descent module and exposed utility extension types for sequential datasets seqdataset and weight vectors weightvector by Peter Prettenhofer
 - Addedpartialfit support for onlineminibatch learning and warmstart to the Stochastic Gradient Descent module by Mathieu Blondel
 - Dense and sparse implementations of Support Vector Machines classes and linearmodel LogisticRegression merged by Lars Buitinck
 - Regressors can now be used as base estimator in the Multiclass and multilabel algorithms module by Mathieu Blondel
 - Added njobs option to metricspairwisepairwisedistances andmetricspairwise pairwisekernels for parallel computation by Mathieu Blondel
 - Kmeans can now be run in parallel using the njobs argument to either Kmeans orKMeans by Robert Layton
 - Improved Crossvalidation evaluating estimator performance andTuning the hyperparameters of an estimator documentation and introduced the new crossvalidationtraintestsplit helper function by Olivier Grisel
 - svmSVC memberscoef andintercept changed sign for consistency with decisionfunction
 - forkernellinear coef was fixed in the onevsone case by Andreas Müller
 - Performance improvements to efficient leaveoneout crossvalidated Ridge regression esp for the nsamples nfeatures case inlinearmodelRidgeCV by Reuben FletcherCostin
 - Refactoring and simplification of the Text feature extraction API (Application Programming Interface) and fixed a bug that caused possible negative IDF by Olivier Grisel
 - Beam pruning option in BaseHMM module has been removed since it is difficult to Cythonize If you are interested in contributing a Cython version you can use the python version in the git history as a reference
 - Classes in Nearest Neighbors now support arbitrary Minkowski metric for nearest neighbors searches The metric can be specified by argument p
- API (Application Programming Interface) changes summary
- covarianceEllipticEnvelop is now deprecated Please use covarianceEllipticEnvelope instead
 - NeighborsClassifier andNeighborsRegressor are gone in the module Nearest Neighbors Use the classes KNeighborsClassifier RadiusNeighborsClassifier KNeighborsRegressor andorRadiusNeighborsRegressor instead
 - Sparse classes in the Stochastic Gradient Descent module are now deprecated

scikitlearn user guide Release 0213

- InmixtureGMM mixtureDPGMM andmixtureVBGMM parameters must be passed to an object when initialising it and not through fit Nowfit will only accept the data as an input parameter
- methods rvs anddecode inGMM module are now deprecated sample andscore orpredict should be used instead
- attributescores andpvalues in univariate feature selection objects are now deprecated scores or pvalues should be used instead
- InLogisticRegression LinearSVC SVC (Support Vector Classification) andNuSVC theclassweight parameter is now an initialization parameter not a parameter to fit This makes grid searches over this parameter possible
- LFWdata is now always shape nsamples nfeatures to be consistent with the Olivetti faces dataset Use images andpairs attribute to access the natural images shapes instead
- InsvmLinearSVC the meaning of the multiclass parameter changed Options now are ovr and crammersinger withovr being the default This does not change the default behavior but hopefully is less confusing
- Class featureselectiontextVectorizer is deprecated and replaced by featureselectiontextTfidfVectorizer
- The preprocessor analyzer nested structure for text feature extraction has been removed All those features are now directly passed as flat constructor arguments to featureselectiontextTfidfVectorizer andfeatureselectiontextCountVectorizer in particular the following parameters are now used
- analyzer can be word or char to switch the default analysis scheme or use a specific python callable as previously
- tokenizer andpreprocessor have been introduced to make it still possible to customize those steps with the new API (Application Programming Interface)
- input explicitly control how to interpret the sequence passed to fit andpredict filenames file objects or direct byte or Unicode strings
- charset decoding is explicit and strict by default
- thevocabulary fitted or not is now stored in the vocabulary attribute to be consistent with the project conventions
- Class featureselectiontextTfidfVectorizer now derives directly from featureselectiontextCountVectorizer to make grid search trivial
- methodsrvs inBaseHMM module are now deprecated sample should be used instead
- Beam pruning option in BaseHMM module is removed since it is difficult to be Cythonized If you are interested you can look in the history codes by git
- The SVMlight format loader now supports files with both zerobased and onebased column indices since both occur “in the wild”
- Arguments in class ShuffleSplit are now consistent with StratifiedShuffleSplit Arguments testfraction andtrainfraction are deprecated and renamed to testsize andtrainsize and can accept both float andint
- Arguments in class Bootstrap are now consistent with StratifiedShuffleSplit Arguments ntest andntrain are deprecated and renamed to testsize andtrainsize and can accept both float andint
- Argument padded to classes in Nearest Neighbors to specify an arbitrary Minkowski metric for nearest neighbors searches

scikitlearn user guide Release 0213

People

- 282 Andreas Müller
- 239 Peter Prettenhofer
- 198 Gael Varoquaux
- 129 Olivier Grisel
- 114 Mathieu Blondel
- 103 Clay Woolam
- 96 Lars Buitinck
- 88 Jaques Grobler
- 82 Alexandre Gramfort
- 50 Bertrand Thirion
- 42 Robert Layton
- 28 flyingimmidev
- 26 Jake Vanderplas
- 26 Shiqiao Du
- 21 Satrajit Ghosh
- 17 David Marek
- 17 Gilles Louppe
- 14 Vlad Niculae
- 11 Yannick Schwartz
- 10 Fabian Pedregosa
- 9 fcostin
- 7 Nick Wilson
- 5 Adrien Gaidon
- 5 Nicolas Pinto
- 4 David WardeFarley
- 5 Nelle Varoquaux
- 5 Emmanuelle Gouillart
- 3 Joonas Sillanpää
- 3 Paolo Losi
- 2 Charles McCarthy
- 2 Roy Hyunjin Han
- 2 Scott White
- 2 ibayer
- 1 Brandyn White
- 1 Carlos Scheidegger

scikitlearn user guide Release 0213

- 1 Claire Revillet
- 1 Conrad Lee
- 1 Edouard Duchesnay
- 1 Jan Hendrik Metzen
- 1 Meng Xinfan
- 1 Rob Zinkov
- 1 Shiqiao
- 1 Udi Weinsberg
- 1 Virgile Fritsch
- 1 Xinfan Meng
- 1 Yaroslav Halchenko
- 1 jansoe
- 1 Leon Palafox

11720 Version 010

January 11 2012

Changelog

- Python 25 compatibility was dropped the minimum Python version needed to use scikitlearn is now 26
- Sparse inverse covariance estimation using the graph Lasso with associated crossvalidated estimator by Gael Varoquaux
- New Tree module by Brian Holt Peter Prettenhofer Satrajit Ghosh and Gilles Louppe The module comes with complete documentation and examples
- Fixed a bug in the RFE module by Gilles Louppe issue 378
- Fixed a memory leak in Support Vector Machines module by Brian Holt issue 367
- Faster tests by Fabian Pedregosa and others
- Silhouette Coefficient cluster analysis evaluation metric added as sklearnmetrics
- silhouettescore by Robert Layton
- Fixed a bug in Kmeans in the handling of the ninit parameter the clustering algorithm used to be run ninit times but the last solution was retained instead of the best solution by Olivier Grisel
- Minor refactoring in Stochastic Gradient Descent module consolidated dense and sparse predict methods Enhanced test time performance by converting model parameters to fortranstyle arrays after fitting only multi class
- Adjusted Mutual Information metric added as sklearnmetricsadjustedmutualinfoscore by Robert Layton
- Models like SVCSVRLinearSVCLogisticRegression from libsvmlliblinear now support scaling of C regularization parameter by the number of samples by Alexandre Gramfort
- New Ensemble Methods module by Gilles Louppe and Brian Holt The module comes with the random forest algorithm and the extratrees method along with documentation and examples

130 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Novelty and Outlier Detection outlier and novelty detection by Virgile Fritsch
- Kernel Approximation a transform implementing kernel approximation for fast SGD on nonlinear kernels by Andreas Müller
- Fixed a bug due to atom swapping in Orthogonal Matching Pursuit OMP by Vlad Niculae
- Sparse coding with a precomputed dictionary by Vlad Niculae
- Mini Batch KMeans performance improvements by Olivier Grisel
- Kmeans support for sparse matrices by Mathieu Blondel
- Improved documentation for developers and for the sklearnutils module by Jake Vanderplas
- Vectorized 20newsgroups dataset loader sklearn.datasets.fetch_20newsgroups_vectorized by Mathieu Blondel
- Multiclass and multilabel algorithms by Lars Buitinck
- Utilities for fast computation of mean and variance for sparse matrices by Mathieu Blondel
- Makes sklearn.preprocessing.scale and sklearn.preprocessingScaler work on sparse matrices by Olivier Grisel
- Feature importances using decision trees and/or forest of trees by Gilles Louppe
- Parallel implementation of forests of randomized trees by Gilles Louppe
- sklearn.cross_validation.ShuffleSplit can subsample the train sets as well as the test sets by Olivier Grisel
- Errors in the build of the documentation fixed by Andreas Müller

API (Application Programming Interface) changes summary

Here are the code migration instructions when upgrading from scikitlearn version 0.9

- Some estimators that may overwrite their inputs to save memory previously had overwrite parameters these have been replaced with copy parameters with exactly the opposite meaning

This particularly affects some of the estimators in linear_model The default behavior is still to copy everything passed in

- The SVMlight dataset loader sklearn.datasets.load_svmlight_file no longer supports loading two files at once use load_svmlight_files instead Also the unused buffer_mb parameter is gone
- Sparse estimators in the Stochastic Gradient Descent module use dense parameter vector coef instead of sparse_coef This significantly improves test time performance
- The Covariance estimation module now has a robust estimator of covariance the Minimum Covariance Determinant estimator
- Cluster evaluation metrics in metrics.cluster have been refactored but the changes are backwards compatible They have been moved to the metrics.cluster.supervised along with metrics.cluster.unsupervised which contains the Silhouette Coefficient
- The permutation_test_score function now behaves the same way as cross_val_score ie uses the mean score across the folds
- Cross Validation generators now use integer indices indices=True by default instead of boolean masks

This makes it more intuitive to use with sparse matrix data

scikitlearn user guide Release 0213

- The functions used for sparse coding `sparseencode` and `sparseencodeparallel` have been combined into `sklearn.decomposition.sparse_encode` and the shapes of the arrays have been transposed for consistency with the matrix factorization setting as opposed to the regression setting
- Fixed an off-by-one error in the `SVMLightLibSVM` file format handling files generated using `sklearn.datasets.dump_svmlight_file` should be regenerated They should continue to work but accidentally had one extra column of zeros prepended
- `BaseDictionaryLearning` class replaced by `SparseCodingMixin`
- `sklearn.util.extmath.fast_svd` has been renamed `sklearn.util.extmath.randomized_svd` and the default oversampling is now fixed to 10 additional random vectors instead of doubling the number of components to extract The new behavior follows the reference paper

People

The following people contributed to scikitlearn since last release

- 246 Andreas Müller
- 242 Olivier Grisel
- 220 Gilles Louppe
- 183 Brian Holt
- 166 Gael Varoquaux
- 144 Lars Buitinck
- 73 Vlad Niculae
- 65 Peter Prettenhofer
- 64 Fabian Pedregosa
- 60 Robert Layton
- 55 Mathieu Blondel
- 52 Jake Vanderplas
- 44 Noel Dawe
- 38 Alexandre Gramfort
- 24 Virgile Fritsch
- 23 Satrajit Ghosh
- 3 Jan Hendrik Metzen
- 3 Kenneth C Arnold
- 3 Shiqiao Du
- 3 Tim SheermanChase
- 3 Yaroslav Halchenko
- 2 Bala Subrahmanyam Varanasi
- 2 DraXus
- 2 Michael Eickenberg
- 1 Bogdan Trach

scikitlearn user guide Release 0213

- 1 FélixAntoine Fortin
- 1 Juan Manuel Caicedo Carvajal
- 1 Nelle Varoquaux
- 1 Nicolas Pinto
- 1 Tiziano Zito
- 1 Xinfan Meng

11721 Version 09

September 21 2011

scikitlearn 09 was released on September 2011 three months after the 08 release and includes the new modules Manifold learning The Dirichlet Process as well as several new algorithms and documentation improvements This release also includes the dictionarylearning work developed by Vlad Niculae as part of the Google Summer of Code program

117 Previous Releases 133

scikitlearn user guide Release 0213

Changelog

- New Manifold learning module by Jake Vanderplas and Fabian Pedregosa
- New Dirichlet Process Gaussian Mixture Model by Alexandre Passos

134 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- Nearest Neighbors module refactoring by Jake Vanderplas general refactoring support for sparse matrices in input speed and documentation improvements See the next section for a full list of API (Application Programming Interface) changes
 - Improvements on the Feature selection module by Gilles Louppe refactoring of the RFE classes documenta tion rewrite increased efficiency and minor API (Application Programming Interface) changes
 - Sparse principal components analysis SparsePCA and MiniBatchSparsePCA by Vlad Niculae Gael Varo quaux and Alexandre Gramfort
 - Printing an estimator now behaves independently of architectures and Python version thanks to Jean Kossaifi
 - Loader for libsvm's svm-light format by Mathieu Blondel and Lars Buitinck
 - Documentation improvements thumbnails in example gallery by Fabian Pedregosa
 - Important bugfixes in Support Vector Machines module segfaults bad performance by Fabian Pedregosa
 - Added Multinomial Naive Bayes and Bernoulli Naive Bayes by Lars Buitinck
 - Text feature extraction optimizations by Lars Buitinck
 - ChiSquare feature selection featureselectionunivariateselectionchi2 by Lars Buit inck
 - Generated datasets module refactoring by Gilles Louppe
 - Multiclass and multilabel algorithms by Mathieu Blondel
 - Ball tree rewrite by Jake Vanderplas
 - Implementation of DBSCAN algorithm by Robert Layton
 - Kmeans predict and transform by Robert Layton
 - Preprocessing module refactoring by Olivier Grisel
 - Faster mean shift by Conrad Lee
 - NewBootstrap Random permutations crossvalidation aka Shuffle Split and various other improve ments in cross validation schemes by Olivier Grisel and Gael Varoquaux
 - Adjusted Rand index and VMeasure clustering evaluation metrics by Olivier Grisel
 - AddedOrthogonal Matching Pursuit by Vlad Niculae
 - Added 2Dpatch extractor utilities in the Feature extraction module by Vlad Niculae
 - Implementation of linearmodelLassoLarsCV crossvalidated Lasso solver using the Lars algorithm andlinearmodelLassoLarsIC BIC/AIC model selection in Lars by Gael Varoquaux and Alexandre Gramfort
 - Scalability improvements to metricsroccurve by Olivier Hervieu
 - Distance helper functions metricspairwisepairwisedistances andmetricspairwise pairwise kernels by Robert Layton
 - MiniBatch KMeans by Nelle Varoquaux and Peter Prettenhofer
 - mldata utilities by Pietro Berkes
 - olivettifaces by David WardeFarley
- API (Application Programming Interface) changes summary
- Here are the code migration instructions when upgrading from scikitlearn version 08
- 117 Previous Releases 135

scikitlearn user guide Release 0213

- Thescikitslearn package was renamed sklearn There is still a scikitslearn package alias for backward compatibility

Thirdparty projects with a dependency on scikitlearn 09 should upgrade their codebase For instance under Linux MacOSX just run make a backup first

find name py xargs sed i s bscikitslearn bsklearn

- Estimators no longer accept model parameters as fit arguments instead all parameters must be only be passed as constructor arguments or using the now public setparams method inherited from base BaseEstimator

Some estimators can still accept keyword arguments on the fit but this is restricted to datadependent values eg a Gram matrix or an affinity matrix that are precomputed from the Xdata matrix

- Thecrossval package has been renamed to crossvalidation although there is also a crossval package alias in place for backward compatibility

Thirdparty projects with a dependency on scikitlearn 09 should upgrade their codebase For instance under Linux MacOSX just run make a backup first

find name py xargs sed i s bcrossval bcrossvalidation

- Thescorefunc argument of the sklearncrossvalidationcrossvalscore function is now expected to accept ytest andypredicted as only arguments for classification and regression tasks orXtest for unsupervised estimators
- gamma parameter for support vector machine algorithms is set to 1 nfeatures by default instead of 1 nsamples

- Thesklearnhmm has been marked as orphaned it will be removed from scikitlearn in version 011 unless someone steps up to contribute documentation examples and fix lurking numerical stability issues

- sklearnneighbors has been made into a submodule The two previously available estimators NeighborsClassifier andNeighborsRegressor have been marked as deprecated Their functionality has been divided among five new classes NearestNeighbors for unsupervised neighbors searches KNeighborsClassifier RadiusNeighborsClassifier for supervised classification problems andKNeighborsRegressor RadiusNeighborsRegressor for supervised regression problems

- sklearnballtreeBallTree has been moved to sklearnneighborsBallTree Using the former will generate a warning

- sklearnlinearmodelLARS and related classes LassoLARS LassoLARSCV etc have been renamed tosklearnlinearmodelLars

- All distance metrics and kernels in sklearnmetricspairwise now have a Y parameter which by default is None If not given the result is the distance or kernel similarity between each sample in Y If given the result is the pairwise distance or kernel similarity between samples in X to Y

- sklearnmetricspairwisel1distance is now called manhattandistance and by default returns the pairwise distance For the component wise distance set the parameter sumoverfeatures to False

Backward compatibility package aliases and other deprecated classes and functions will be removed in version 011

People

38 people contributed to this release

- 387 Vlad Niculae

136 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- 320 Olivier Grisel
 - 192 Lars Buitinck
 - 179 Gael Varoquaux
 - 168 Fabian Pedregosa INRIA Parietal Team
 - 127 Jake Vanderplas
 - 120 Mathieu Blondel
 - 85 Alexandre Passos
 - 67 Alexandre Gramfort
 - 57 Peter Prettenhofer
 - 56 Gilles Louppe
 - 42 Robert Layton
 - 38 Nelle Varoquaux
 - 32 Jean Kossaifi
 - 30 Conrad Lee
 - 22 Pietro Berkes
 - 18 andy
 - 17 David WardeFarley
 - 12 Brian Holt
 - 11 Robert
 - 8 Amit Aides
 - 8 Virgile Fritsch
 - 7 Yaroslav Halchenko
 - 6 Salvatore Masecchia
 - 5 Paolo Losi
 - 4 Vincent Schut
 - 3 Alexis Metaireau
 - 3 Bryan Silverthorn
 - 3 Andreas Müller
 - 2 Minwoo Jake Lee
 - 1 Emmanuelle Gouillart
 - 1 Keith Goodman
 - 1 Lucas Wiman
 - 1 Nicolas Pinto
 - 1 Thouis Ray Jones
 - 1 Tim SheermanChase
- 117 Previous Releases 137

scikitlearn user guide Release 0213

11722 Version 08

May 11 2011

scikitlearn 08 was released on May 2011 one month after the first “international” scikitlearn coding sprint and is marked by the inclusion of important modules Hierarchical clustering Cross decomposition Nonnegative matrix factorization NMF (Non-Negative Matrix Factorization) or NMF initial support for Python 3 and by important enhancements and bug fixes

Changelog

Several new modules were introduced during this release

- New Hierarchical clustering module by Vincent Michel Bertrand Thirion Alexandre Gramfort and Gael Varo

quaux

- Kernel PCA (Principal Component Analysis) implementation by Mathieu Blondel

- labeledfacesinthewild by Olivier Grisel

- New Cross decomposition module by Edouard Duchesnay

- Nonnegative matrix factorization NMF (Non-Negative Matrix Factorization) or NMF module Vlad Niculae

- Implementation of the Oracle Approximating Shrinkage algorithm by Virgile Fritsch in the Covariance estima

tionmodule

Some other modules benefited from significant improvements or cleanups

- Initial support for Python 3 builds and imports cleanly some modules are usable while others have failing tests

by Fabian Pedregosa

- decompositionPCA is now usable from the Pipeline object by Olivier Grisel

- Guide How to optimize for speed by Olivier Grisel

- Fixes for memory leaks in libsvm bindings 64bit safer BallTree by Lars Buitinck

- bug and style fixing in Kmeans algorithm by Jan Schlüter

- Add attribute converged to Gaussian Mixture Models by Vincent Schut

- Implemented transform predictlogproba indiscriminantanalysis

LinearDiscriminantAnalysis By Mathieu Blondel

- Refactoring in the Support Vector Machines module and bug fixes by Fabian Pedregosa Gael Varoquaux and

Amit Aides

- Refactored SGD module removed code duplication better variable naming added interface for sample weight

by Peter Prettenhofer

- Wrapped BallTree with Cython by Thouis Ray Jones

- Added function svmll1minc by Paolo Losi

- Typos doc style etc by Yaroslav Halchenko Gael Varoquaux Olivier Grisel Yann Malet Nicolas Pinto Lars

Buitinck and Fabian Pedregosa

People

People that made this release possible preceded by number of commits

- 159 Olivier Grisel

138 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

- 96 Gael Varoquaux
- 96 Vlad Niculae
- 94 Fabian Pedregosa
- 36 Alexandre Gramfort
- 32 Paolo Losi
- 31 Edouard Duchesnay
- 30 Mathieu Blondel
- 25 Peter Prettenhofer
- 22 Nicolas Pinto
- 11 Virgile Fritsch
- 7 Lars Buitinck
- 6 Vincent Michel
- 5 Bertrand Thirion
- 4 Thouis Ray Jones
- 4 Vincent Schut
- 3 Jan Schlüter
- 2 Julien Miotte
- 2 Matthieu Perrot
- 2 Yann Malet
- 2 Yaroslav Halchenko
- 1 Amit Aides
- 1 Andreas Müller
- 1 Feth Arezki
- 1 Meng Xinfan

11723 Version 07

March 2 2011

scikitlearn 07 was released in March 2011 roughly three months after the 06 release This release is marked by the speed improvements in existing algorithms like kNearest Neighbors and KMeans algorithm and by the inclusion of an efficient algorithm for computing the Ridge Generalized Cross Validation solution Unlike the preceding release no new modules were added to this release

Changelog

- Performance improvements for Gaussian Mixture Model sampling Jan Schlüter
- Implementation of efficient leaveoneout crossvalidated Ridge in linearmodelRidgeCV Mathieu Blondel

117 Previous Releases 139

scikitlearn user guide Release 0213

- Better handling of collinearity and early stopping in linearmodellarspath Alexandre Gramfort and Fabian Pedregosa
- Fixes for liblinear ordering of labels and sign of coefficients Dan Yamins Paolo Losi Mathieu Blondel and Fabian Pedregosa
- Performance improvements for Nearest Neighbors algorithm in highdimensional spaces Fabian Pedregosa
- Performance improvements for clusterKMeans Gael Varoquaux and James Bergstra
- Sanity checks for SVMbased classes Mathieu Blondel
- Refactoring of neighborsNeighborsClassifier andneighborskneighborsgraph added different algorithms for the kNearest Neighbor Search and implemented a more stable algorithm for finding barycenter weights Also added some developer documentation for this module see notesneighbors for more information Fabian Pedregosa
- Documentation improvements Added pcaRandomizedPCA andlinearmodel LogisticRegression to the class reference Also added references of matrices used for clustering and other fixes Gael Varoquaux Fabian Pedregosa Mathieu Blondel Olivier Grisel Virgile Fritsch Emmanuelle Gouillart
- Binded decisionfunction in classes that make use of liblinear dense and sparse variants like svm LinearSVC orlinearmodelLogisticRegression Fabian Pedregosa
- Performance and API (Application Programming Interface) improvements to metricseuclideandistances and topca RandomizedPCA James Bergstra
- Fix compilation issues under NetBSD Kamel Ibn Hassen Derouiche
- Allow input sequences of different lengths in hmmGaussianHMM Ron Weiss
- Fix bug in affinity propagation caused by incorrect indexing Xinfan Meng

People that made this release possible preceded by number of commits

- 85 Fabian Pedregosa
- 67 Mathieu Blondel
- 20 Alexandre Gramfort
- 19 James Bergstra
- 14 Dan Yamins
- 13 Olivier Grisel
- 12 Gael Varoquaux
- 4 Edouard Duchesnay
- 4 Ron Weiss
- 2 Satrajit Ghosh
- 2 Vincent Dubourg
- 1 Emmanuelle Gouillart
- 1 Kamel Ibn Hassen Derouiche
- 1 Paolo Losi

scikitlearn user guide Release 0213

- 1 VirgileFritsch
- 1 Yaroslav Halchenko
- 1 Xinfan Meng

11724 Version 06

December 21 2010

scikitlearn 06 was released on December 2010 It is marked by the inclusion of several new modules and a general renaming of old ones It is also marked by the inclusion of new example including applications to realworld datasets

- Changelog
- New stochastic gradient descent module by Peter Prettenhofer The module comes with complete documentation and examples
 - Improved svm module memory consumption has been reduced by 50 heuristic to automatically set class weights possibility to assign weights to samples see SVM (Support Vector Machine) Weighted samples for an example
 - New Gaussian Processes module by Vincent Dubourg This module also has great documentation and some very neat examples See `examplegaussianprocessplotgpregressionpy` or `examplegaussianprocessplotgpprobabilisticclassificationafterregressionpy` for a taste of what can be done
 - It is now possible to use liblinear's Multiclass SVC (Support Vector Classification) option `multiclass` in `svmLinearSVC`
 - New features and performance improvements of text feature extraction
 - Improved sparse matrix support both in main classes `gridsearchGridSearchCV` as in modules `sklearnsvmsparse` and `sklearnlinearmodelsparse`
 - Lots of cool new examples and a new section that uses realworld datasets was created These include Faces recognition example using eigenfaces and SVMs Species distribution modeling Libsvm GUI Wikipedia principal eigenvector and others
 - Faster Least Angle Regression algorithm It is now 2x faster than the R version on worst case and up to 10x times faster on some cases
 - Faster coordinate descent algorithm In particular the full path version of `lasso linearmodel lassopath` is more than 200x times faster than before
 - It is now possible to get probability estimates from a `linearmodelLogisticRegression` model
 - module renaming the `glm` module has been renamed to `linearmodel` the `gmm` module has been included into the more general mixture model and the `sgd` module has been included in `linearmodel`
 - Lots of bug fixes and documentation improvements

People

People that made this release possible preceded by number of commits

- 207 Olivier Grisel
 - 167 Fabian Pedregosa
 - 97 Peter Prettenhofer
 - 68 Alexandre Gramfort
- 117 Previous Releases 141

scikitlearn user guide Release 0213

- 59 Mathieu Blondel
- 55 Gael Varoquaux
- 33 Vincent Dubourg
- 21 Ron Weiss
- 9 Bertrand Thirion
- 3 Alexandre Passos
- 3 AnneLaure Fouque
- 2 Ronan Amicel
- 1 Christian Osendorfer

11725 Version 05

October 11 2010

Changelog

New classes

- Support for sparse matrices in some classifiers of modules svm and linear model see `sparseSVC` `svmSVC` `sparseSVR` `svmSVC` `sparseLinearSVC` `linearModelSVC` `sparseLasso` `linearModelSVC` `sparseElasticNet`

- New `pipeline` object to compose different estimators

- Recursive Feature Elimination routines in module `Feature selection`

- Addition of various classes capable of cross validation in the `linear model` module `linear model` `LassoCV` `linear model` `ElasticNetCV` etc

- New more efficient LARS algorithm implementation The Lasso variant of the algorithm is also implemented

See `linear model` `lars` `lars path` `linear model` `Lars` and `linear model` `LassoLars`

- New Hidden Markov Models module see classes `hmmGaussianHMM` `hmmMultinomialHMM` `hmmGMMHMM`

- New module `feature extraction` see class reference

- New `FastICA` algorithm in module `sklearnfastica`

Documentation

- Improved documentation for many modules now separating narrative documentation from the class reference

As an example see documentation for the SVM (Support Vector Machine) module and the complete class reference

Fixes

- API (Application Programming Interface) changes adhere variable names to PEP8 give more meaningful names
- Fixes for svm module to run on a shared memory context multiprocessing
- It is again possible to generate latex and thus PDF from the sphinx docs

142 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

Examples

- new examples using some of the mlcomp datasets sphxglrautoexamplesmlcompsparseddocumentclassification
- pysince removed and Classification of text documents using sparse features
- Many more examples See here the full list of examples

External dependencies

- Joblib is now a dependency of this package although it is shipped with sklearnexternalsjoblib

Removed modules

- Module ann Artificial Neural Networks has been removed from the distribution Users wanting this sort of algorithms should take a look into pybrain

Misc

- New sphinx theme for the web page

Authors

The following is a list of authors for this release preceded by number of commits

- 262 Fabian Pedregosa
- 240 Gael Varoquaux
- 149 Alexandre Gramfort
- 116 Olivier Grisel
- 40 Vincent Michel
- 38 Ron Weiss
- 23 Matthieu Perrot
- 10 Bertrand Thirion
- 7 Yaroslav Halchenko
- 9 VirgileFritsch
- 6 Edouard Duchesnay
- 4 Mathieu Blondel
- 1 Ariel Rokem
- 1 Matthieu Brucher

11726 Version 04

August 26 2010

117 Previous Releases 143

scikitlearn user guide Release 0213

Changelog

Major changes in this release include

- Coordinate Descent algorithm Lasso ElasticNet refactoring speed improvements roughly 100x times faster
- Coordinate Descent Refactoring and bug fixing for consistency with R's package GLMNET
- New metrics module
- New GMM module contributed by Ron Weiss
- Implementation of the LARS algorithm without Lasso variant for now
- featureselection module redesign
- Migration to GIT as version control system
- Removal of obsolete attrselect module
- Rename of private compiled extensions added underscore
- Removal of legacy unmaintained code
- Documentation improvements both docstring and rst
- Improvement of the build system to optionally link with MKL Also provide a lite BLAS implementation in case no systemwide BLAS is found
- Lots of new examples
- Many many bug fixes

Authors

The committer list for this release is the following preceded by number of commits

- 143 Fabian Pedregosa
- 35 Alexandre Gramfort
- 34 Olivier Grisel
- 11 Gael Varoquaux
- 5 Yaroslav Halchenko
- 2 Vincent Michel
- 1 Chris Filo Gorgolewski

11727 Earlier versions

Earlier versions included contributions by Fred Mailhot David Cooke David Huard Dave Morrill Ed Schofield

Travis Oliphant Pearu Peterson

144 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

118 Roadmap

1181 Purpose of this document

This document list general directions that core contributors are interested to see developed in scikitlearn The fact that an item is listed here is in no way a promise that it will happen as resources are limited Rather it is an indication that help is welcomed on this topic

1182 Statement of purpose Scikitlearn in 2018

Eleven years after the inception of Scikitlearn much has changed in the world of machine learning Key changes include

- Computational tools The exploitation of GPUs distributed programming frameworks like ScalaSpark etc
- Highlevel Python libraries for experimentation processing and data management Jupyter notebook Cython Pandas Dask Numba

• Changes in the focus of machine learning research artificial intelligence applications where input structure is key with deep learning representation learning reinforcement learning domain transfer etc

A more subtle change over the last decade is that due to changing interests in ML PhD students in machine learning are more likely to contribute to PyTorch Dask etc than to Scikitlearn so our contributor pool is very different to a decade ago

Scikitlearn remains very popular in practice for trying out canonical machine learning techniques particularly for applications in experimental science and in data science A lot of what we provide is now very mature But it can be costly to maintain and we cannot therefore include arbitrary new implementations Yet Scikitlearn is also essential in defining an API (Application Programming Interface) framework for the development of interoperable machine learning components external to the library

Thus our main goals in this era are to

- continue maintaining a highquality welldocumented collection of canonical tools for data processing and machine learning within the current scope ie rectangular data largely invariant to column and row order predicting targets with simple structure
- improve the ease for users to develop and publish external components
- improve interoperability with modern data science tools eg Pandas Dask and infrastructures eg distributed processing

Many of the more finegrained goals can be found under the API (Application Programming Interface) tag on the issue tracker

1183 Architectural general goals

The list is numbered not as an indication of the order of priority but to make referring to specific points easier Please add new entries only at the bottom

1 Everything in Scikitlearn should conform to our API (Application Programming Interface) contract

- Pipeline and FeatureUnion modify their input parameters in fit Fixing this requires making sure we have a good grasp of their use cases to make sure all current functionality is maintained 8157 7382

2 Improved handling of Pandas DataFrames and SparseDataFrames

- document current handling
- column reordering issue 7242

118 Roadmap 145

scikitlearn user guide Release 0213

- avoiding unnecessary conversion to ndarray
- returning DataFrames from transformers 5523
- getting DataFrames from dataset loaders
- Sparse currently not considered

3 Improved handling of categorical features

- Treebased models should be able to handle both continuous and categorical features 4899
- In dataset loaders
- As generic transformers to be used with ColumnTransforms eg ordinal encoding supervised by correlation with target variable

4 Improved handling of missing data

- Making sure metaestimators are lenient towards missing data
- Nontrivial imputers
- Learners directly handling missing data
- An amputation sample generator to make parts of a dataset go missing
- Handling mixtures of categorical and continuous variables

5 Passing around information that is not X y Sample properties

- We need to be able to pass sample weights to scorers in cross validation
- We should have standard generalised ways of passing samplewise properties around in metaestimators 4497 7646

6 Passing around information that is not X y Feature properties

- Feature names or descriptions should ideally be available to fit for eg 6425 6424
- Feature handling eg “is this a nominal ordinal English language text” should also not need to be provided to estimator constructors ideally but should be available as metadata alongside X 8480

7 Passing around information that is not X y Target information

- We have problems getting the full set of classes to all components when the data is split sampled 6231 8100

- We have no way to handle a mixture of categorical and continuous targets

8 Make it easier for external users to write Scikitlearncompatible components

- More flexible estimator checks that do not select by estimator name 6599 6715
- Example of how to develop a metaestimator
- More selfsufficient running of scikitlearncontrib or a similar resource

9 Support resampling and sample reduction

- Allow subsampling of majority classes in a pipeline 3855
- Implement random forests with resampling 8732

10 Better interfaces for interactive development

- repr and HTML visualisations of estimators 6323
- Include plotting tools not just as examples 9173

146 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

11 Improved tools for model diagnostics and basic inference

- alternative feature importances implementations eg methods or wrappers
- better ways to handle validation sets when fitting
- better ways to find thresholds create decision rules 8614

12 Better tools for selecting hyperparameters with transductive estimators

- Grid search and cross validation are not applicable to most clustering tasks Stabilitybased selection is more relevant

13 Improved tracking of fitting

- Verbose is not very friendly and should use a standard logging library 6929
- Callbacks or a similar system would facilitate logging and early stopping

14 Distributed parallelism

- Joblib can now plug onto several backends some of them can distribute the computation across computers
- However we want to stay high level in scikitlearn

15 A way forward for more out of core

- Dask enables easy outofcore computation While the dask model probably cannot be adaptable to all machinelearning algorithms most machine learning is on smaller data than ETL hence we can maybe adapt to very large scale while supporting only a fraction of the patterns

16 Better support for manual and automatic pipeline building

- Easier way to construct complex pipelines and valid search spaces 7608 5082 8243
- provide search ranges for common estimators
- cf searchgrid

17 Support for working with pretrained models

- Estimator “freezing” In particular right now it’s impossible to clone a CalibratedClassifierCV with prefit 8370 6451

18 Backwardscompatible deserialization of some estimators

- Currently serialization with pickle breaks across versions While we may not be able to get around other limitations of pickle re security etc it would be great to offer crossversion safety from version 10 Note Gael and Olivier think that this can cause heavy maintenance burden and we should manage the tradeoffs A possible alternative is presented in the following point

19 Documentation and tooling for model lifecycle management

- Document good practices for model deployments and lifecycle before deploying a model snapshot the code versions numpy scipy scikitlearn custom code repo the training script and an alias on how to retrieve historical training data snapshot a copy of a small validation set snapshot of the predictions predicted probabilities for classifiers on that validation set

- Document and tools to make it easy to manage upgrade of scikitlearn versions

-Try to load the old pickle if it works use the validation set prediction snapshot to detect that the serialized model still behave the same

-If joblibload pickleload not work use the versioned control training script historical training set to retrain the model and use the validation set prediction snapshot to assert that it is possible to recover the previous predictive performance if this is not the case there is probably a bug in scikitlearn that needs to be reported

118 Roadmap 147

scikitlearn user guide Release 0213

20 Optional Improve scikitlearn common tests suite to make sure that at least for frequently used models have stable predictions across versions to be discussed

- Extend documentation to mention how to deploy models in Python-free environments for instance ONNX and use the above best practices to assess predictive consistency between scikitlearn and ONNX prediction functions on validation set
- Document good practices to detect temporal distribution drift for deployed model and good practices for retraining on fresh data without causing catastrophic predictive performance regressions

21 More didactic documentation

- More and more options have been added to scikitlearn As a result the documentation is crowded which makes it hard for beginners to get the big picture Some work could be done in prioritizing the information

1184 Subpackage-specific goals

sklearncluster

- kmeans variants for non-Euclidean distances if we can show these have benefits beyond hierarchical clustering

sklearnensemble

- a stacking implementation

sklearnmodelselection

- multimetric scoring is slow 9326
- perhaps we want to be able to get back more than multiple metrics
- the handling of random states in CV (Cross-Validation) splitters is a poor design and contradicts the validation of similar parameters in estimators

- exploit warmstarting and path algorithms so the benefits of EstimatorCV objects can be accessed via

GridSearchCV and used in Pipelines 1626

- Crossvalidation should be able to be replaced by OOB estimates whenever a crossvalidation iterator is used
- Redundant computations in pipelines should be avoided related to point above cf daskml

sklearnneighbors

- Ability to substitute a custom approximate precomputed nearest neighbors implementation for ours in almost all contexts that nearest neighbors are used for learning 10463

sklearnpipeline

- Performance issues with Pipelinememory
- see “Everything in Scikitlearn should conform to our API (Application Programming Interface) contract” above

119 Scikitlearn governance and decisionmaking

The purpose of this document is to formalize the governance process used by the scikitlearn project to clarify how decisions are made and how the various elements of our community interact This document establishes a decision making structure that takes into account feedback from all members of the community and strives to find consensus while avoiding any deadlocks

148 Chapter 1 Welcome to scikitlearn

scikitlearn user guide Release 0213

This is a meritocratic consensusbased community project Anyone with an interest in the project can join the community contribute to the project design and participate in the decision making process This document describes how that participation takes place and how to set about earning merit within the project community

1191 Roles And Responsibilities

Contributors

Contributors are community members who contribute in concrete ways to the project Anyone can become a contributor and contributions can take many forms – not only code – as detailed in the contributors guide

Core developers

Core developers are community members who have shown that they are dedicated to the continued development of the project through ongoing engagement with the community They have shown they can be trusted to maintain Scikit learn with care Being a core developer allows contributors to more easily carry on with their project related activities by giving them direct access to the project’s repository and is represented as being an organization member on the scikitlearn GitHub organization Core developers are expected to review code contributions can merge approved pull requests can cast votes for and against merging a pullrequest and can be involved in deciding major changes to the API (Application Programming Interface)

New core developers can be nominated by any existing core developers Once they have been nominated there will be a vote by the current core developers Voting on new core developers is one of the few activities that takes place on the project’s private management list While it is expected that most votes will be unanimous a twothirds majority of the cast votes is enough The vote needs to be open for at least 1 week

Core developers that have not contributed to the project commits or GitHub comments in the past 12 months will be asked if they want to become emeritus core developers and recant their commit and voting rights until they become active again The list of core developers active and emeritus with dates at which they became active is public on the scikitlearn website

Technical Committee

The Technical Committee TC members are core developers who have additional responsibilities to ensure the smooth running of the project TC members are expected to participate in strategic planning and approve changes to the governance model The purpose of the TC is to ensure a smooth progress from the bigpicture perspective Indeed changes that impact the full project require a synthetic analysis and a consensus that is both explicit and informed In cases that the core developer community which includes the TC members fails to reach such a consensus in the required time frame the TC is the entity to resolve the issue Membership of the TC is by nomination by a core developer A nomination will result in discussion which cannot take more than a month and then a vote by the core developers which will stay open for a week TC membership votes are subject to a twothird majority of all cast votes as well as a simple majority approval of all the current TC members TC members who do not actively engage with the TC duties are expected to resign

The initial Technical Committee of scikitlearn consists of Alexandre Gramfort Olivier Grisel Andreas Müller Joel Nothman Hanmin Qin Gaël Varoquaux and Roman Yurchak

1192 Decision Making Process

Decisions about the future of the project are made through discussion with all members of the community All non sensitive project management discussion takes place on the project contributors’ mailing list and the issue tracker Occasionally sensitive discussion occurs on a private list

119 Scikitlearn governance and decisionmaking 149

scikitlearn user guide Release 0213

Scikitlearn uses a “consensus seeking” process for making decisions The group tries to find a resolution that has no open objections among core developers At any point during the discussion any coredeveloper can call for a vote which will conclude one month from the call for the vote Any vote must be backed by a SLEP If no option can gather two thirds of the votes cast the decision is escalated to the TC which in turn will use consensus seeking with the fallback option of a simple majority vote if no consensus can be found within a month This is what we hereafter may refer to as “the decision making process”

Decisions in addition to adding core developers and TC membership as above are made according to the following rules

- Minor Documentation changes such as typo fixes or addition correction of a sentence but no change of the scikitlearnorg landing page or the “about” page Requires 1 by a core developer no 1 by a core developer lazy consensus happens on the issue or pull request page Core developers are expected to give “reasonable time” to others to give their opinion on the pull request if they’re not confident others would agree
- Code changes and major documentation changes require 1 by two core developers no 1 by a core developer lazy consensus happens on the issue of pullrequest page
- Changes to the API (Application Programming Interface) principles and changes to dependencies or supported versions happen via a Enhancement proposals SLEPs and follows the decisionmaking process outlined above
- Changes to the governance model use the same decision process outlined above

If a veto 1 vote is cast on a lazy consensus the proposer can appeal to the community and core developers and the change can be approved or rejected using the decision making procedure outlined above

1193 Enhancement proposals SLEPs

For all votes a proposal must have been made public and discussed before the vote Such proposal must be a consolidated document in the form of a ‘ScikitLearn Enhancement Proposal’ SLEP rather than a long discussion on an issue A SLEP must be submitted as a pullrequest to enhancement proposals using the SLEP template

CHAPTER
TWO
SCIKITLEARN TUTORIALS

21 An introduction to machine learning with scikitlearn

Section contents

In this section we introduce the machine learning vocabulary that we use throughout scikitlearn and give a simple learning example

211 Machine learning the problem setting

In general a learning problem considers a set of n samples of data and then tries to predict properties of unknown data. If each sample is more than a single number and for instance a multidimensional entry aka multivariate data it is said to have several attributes or features

Learning problems fall into a few categories

- supervised learning in which the data comes with additional attributes that we want to predict. Click here to go to the scikitlearn supervised learning page

This problem can be either -classification samples belong to two or more classes and we want to learn from already labeled data how to predict the class of unlabeled data. An example of a classification problem would be handwritten digit recognition in which the aim is to assign each input vector to one of a finite number of discrete categories

Another way to think of classification is as a discrete as opposed to continuous form of supervised learning where one has a limited number of categories and for each of the n samples provided one is to try to label them with the correct category or class

-regression if the desired output consists of one or more continuous variables then the task is called regression. An example of a regression problem would be the prediction of the length of a salmon as a function of its age and weight

- unsupervised learning in which the training data consists of a set of input vectors x without any corresponding target values. The goal in such problems may be to discover groups of similar examples within the data where it is called clustering or to determine the distribution of data within the input space known as density estimation or to project the data from a highdimensional space down to two or three dimensions for the purpose of visualization. Click here to go to the ScikitLearn unsupervised learning page

scikitlearn user guide Release 0213

Training set and testing set

Machine learning is about learning some properties of a data set and then testing those properties against another data set A common practice in machine learning is to evaluate an algorithm by splitting a data set into two We call one of those sets the training set on which we learn some properties we call the other set the testing set on which we test the learned properties

212 Loading an example dataset

scikitlearn comes with a few standard datasets for instance the iris and digits datasets for classification and the boston house prices dataset for regression

In the following we start a Python interpreter from our shell and then load the iris and digits datasets Our notational convention is that `!` denotes the shell prompt while `>` denotes the Python interpreter prompt

```
python
from sklearn import datasets
iris = datasets.load_iris()
digits = datasets.load_digits()
```

A dataset is a dictionarylike object that holds all the data and some metadata about the data This data is stored in the `data` member which is a `(n_samples, n_features)` array In the case of supervised problem one or more response variables are stored in the `target` member More details on the different datasets can be found in the dedicated section

For instance in the case of the digits dataset `digits.data` gives access to the features that can be used to classify the digits samples

```
print(digits.data)
0 0 5 0 0 0
0 0 0 10 0 0
0 0 0 16 9 0
```

```
0 0 1 6 0 0
0 0 2 12 0 0
0 0 10 12 1 0
```

`digits.target` gives the ground truth for the digit dataset that is the number corresponding to each digit image that we are trying to learn

```
digits.target
array(0 1 2 8 9 8)
```

Shape of the data arrays

The data is always a 2D array shape `(n_samples, n_features)` although the original data may have had a different shape In the case of the digits each original sample is an image of shape `(8, 8)` and can be accessed using

```
digits.images[0]
array([[0, 0, 5, 13, 9, 1, 0, 0],
       [0, 0, 13, 15, 10, 15, 5, 0],
       [0, 3, 15, 2, 0, 11, 8, 0],
       [0, 4, 12, 0, 0, 8, 8, 0],
       [0, 5, 8, 0, 0, 9, 8, 0],
       [0, 4, 11, 0, 1, 12, 7, 0],
       [0, 2, 14, 5, 10, 12, 0, 0],
       [0, 0, 6, 13, 10, 0, 0, 0]])
```

scikitlearn user guide Release 0213

This simple example on this dataset illustrates how starting from the original problem one can shape the data for consumption in scikitlearn

Loading from external datasets

To load from an external dataset please refer to loading external datasets

213 Learning and predicting

In the case of the digits dataset the task is to predict given an image which digit it represents We are given samples of each of the 10 possible classes the digits zero through nine on which we fit an estimator to be able to predict the classes to which unseen samples belong

In scikitlearn an estimator for classification is a Python object that implements the methods fit(X, y) and predict()

An example of an estimator is the class sklearn.svm.SVC which implements support vector classification The estimator's constructor takes as arguments the model's parameters

For now we will consider the estimator as a black box

```
from sklearn import svm
clf = svm.SVC(gamma=0.001, C=100)
```

Choosing the parameters of the model

In this example we set the value of gamma manually To find good values for these parameters we can use tools such as grid search and cross validation

The clf for classifier estimator instance is first fitted to the model that is it must learn from the model This is done by passing our training set to the fit method For the training set we'll use all the images from our dataset except for the last image which we'll reserve for our predicting We select the training set with the 1 Python syntax which produces a new array that contains all but the last item from digitsdata

```
clf.fit(digitsdata[:-1], digitstarget[:-1])
SVCC1000, cachesize=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
max_iter=1, probability=False, random_state=None, shrinking=True,
tol=0.0001, verbose=False
```

Now you can predict new values In this case you'll predict using the last image from digitsdata By predicting you'll determine the image from the training set that best matches the last image

```
clf.predict(digitsdata[-1])
array(8)
```

21 An introduction to machine learning with scikitlearn 153

scikitlearn user guide Release 0213

The corresponding image is

As you can see it is a challenging task after all the images are of poor resolution Do you agree with the classifier

A complete example of this classification problem is available as an example that you can run and study Recognizing handwritten digits

214 Model persistence

It is possible to save a model in scikitlearn by using Python’s builtin persistence model pickle

```
from sklearn import svm
from sklearn import datasets
clf = svmSVC(gamma=scale
iris = datasets.load_iris
X, y = iris.data, iris.target
clf.fit(X, y)
SVCC10, cache_size=200, class_weight=None, coef0=0.0
decision_function_shape='ovr', degree=3, gamma=scale, kernel='rbf'
max_iter=1, probability=False, random_state=None, shrinking=True
tol=0.0001, verbose=False
import pickle
s = pickle.dumps(clf)
clf2 = pickle.loads(s)
clf2.predict(X01)
array(0)
y0
0
```

In the specific case of scikitlearn it may be more interesting to use joblib’s replacement for pickle joblib.dump joblib.load which is more efficient on big data but it can only pickle to the disk and not to a string

```
from joblib import dump, load
dump(clf, filename='joblib')
```

Later you can reload the pickled model possibly in another Python process with

```
clf = load(filename='joblib')
```

Note joblib.dump and joblib.load functions also accept filelike object instead of filenames More information on data persistence with Joblib is available here

Note that pickle has some security and maintainability issues Please refer to section Model persistence for more detailed information about model persistence with scikitlearn

scikitlearn user guide Release 0213

215 Conventions

scikitlearn estimators follow certain rules to make their behavior more predictive These are described in more detail in the Glossary of Common Terms and API (Application Programming Interface) Elements

Type casting

Unless otherwise specified input will be cast to float64

```
import numpy as np
from sklearn import randomprojection
rng = np.random.RandomState(0)
X = rng.rand(10, 2000)
X = np.array(X, dtype=float32)
X.dtype
dtypefloat32
transformer = randomprojection.GaussianRandomProjection()
X_new = transformer.fit_transform(X)
X_new.dtype
dtypefloat64
```

In this example X is float32 which is cast to float64 by fit_transform(X)

Regression targets are cast to float64 and classification targets are maintained

```
from sklearn import datasets
from sklearn.svm import SVC (Support Vector Classification)
iris = datasets.load_iris()
clf = SVC(gamma=scale)
clf.fit(iris.data, iris.target)
SVCC10 cachesize=200 classweight=None coef=0.00
decisionfunctionshape=ovr degree=3 gamma=scale kernel=rbf
max_iter=1 probability=False randomstate=None shrinking=True
tol=0.001 verbose=False
list(clf.predict(iris.data))
[0, 0, 0]
clf.fit(iris.data, iris.target_names[iris.target])
SVCC10 cachesize=200 classweight=None coef=0.00
decisionfunctionshape=ovr degree=3 gamma=scale kernel=rbf
max_iter=1 probability=False randomstate=None shrinking=True
tol=0.001 verbose=False
list(clf.predict(iris.data))
setosa setosa setosa
```

Here the first predict returns an integer array since iris.target an integer array was used in fit The second predict returns a string array since iris.target_names was for fitting

Refitting and updating parameters

Hyperparameters of an estimator can be updated after it has been constructed via the set_params method Calling fit more than once will overwrite what was learned by any previous fit

scikitlearn user guide Release 0213

```
import numpy as np
from sklearn.datasets import loadiris
from sklearn.svm import SVC (Support Vector Classification)
X, y = loadiris(returnXy=True)
clf = SVC (Support Vector Classification)
clf.set_params(kernel='linear', fit_X_y=
SVCC10, cache_size=200, class_weight=None, coef=0.00
decision_function_shape='ovr', degree=3, gamma='auto', deprecated
kernel='linear', max_iter=1, probability=False, random_state=None
shrinking=True, tol=0.0001, verbose=False)
clf.predict(X5)
array([0, 0, 0, 0, 0])
clf.set_params(kernel='rbf', gamma='scale', fit_X_y=
SVCC10, cache_size=200, class_weight=None, coef=0.00
decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf'
max_iter=1, probability=False, random_state=None, shrinking=True
tol=0.0001, verbose=False)
clf.predict(X5)
array([0, 0, 0, 0, 0])
```

Here the default kernel 'rbf' is first changed to 'linear' via `SVC.set_params` after the estimator has been constructed and changed back to 'rbf' to refit the estimator and to make a second prediction.

Multiclass vs multilabel fitting

When using multiclass classifiers, the learning and prediction task that is performed is dependent on the format of the target data fit upon.

```
from sklearn.svm import SVC (Support Vector Classification)
from sklearn.multiclass import OneVsRestClassifier
from sklearn.preprocessing import LabelBinarizer
X = [1, 2, 2, 4, 4, 5, 3, 2, 3, 1]
y = [0, 0, 1, 1, 2]
classifier = OneVsRestClassifier(estimator=SVC(gamma='scale',
random_state=0))
classifier.fit(X, y)
classifier.predict(X)
array([0, 0, 1, 1, 2])
```

In the above case, the classifier is fit on a 1d array of multiclass labels and the `predict` method therefore provides corresponding multiclass predictions. It is also possible to fit upon a 2d array of binary label indicators.

```
y = LabelBinarizer().fit_transform(y)
classifier.fit(X, y)
classifier.predict(X)
array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 0],
       [0, 0, 0]])
```

Here the classifier is fit on a 2d binary label representation of `y` using the `LabelBinarizer`. In this case, `predict` returns a 2d array representing the corresponding multilabel predictions.

scikitlearn user guide Release 0213

Note that the fourth and fifth instances returned all zeroes indicating that they matched none of the three labels fit upon With multilabel outputs it is similarly possible for an instance to be assigned multiple labels

```
from sklearn.preprocessing import MultiLabelBinarizer
```

```
y = [0, 1, 0, 2, 1, 3, 0, 2, 3, 2, 4]
```

```
mlb = MultiLabelBinarizer(fit_transformer)
```

```
mlb.fit(X, y)
```

```
array([[1, 0, 0, 0]
```

```
       [0, 1, 0, 0]
```

```
       [0, 1, 0, 1]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

```
       [1, 0, 1, 0]
```

scikitlearn user guide Release 0213

When the data is not initially in the nsamples nfeatures shape it needs to be preprocessed in order to be used by scikitlearn

An example of reshaping data would be the digits dataset

The digits dataset is made of 1797 8x8 images of handwritten digits

digits datasetsloadaddigits

digitsimagesshape

1797 8 8

import matplotlib.pyplot as plt

plt.imshow(digitsimages[1], cmap=plt.cm.gray)

matplotlib.image.AxesImage object at

To use this dataset with scikitlearn we transform each 8x8 image into a feature vector of length 64

data = digitsimages.reshape(digitsimages.shape[0], 1)

Estimators objects

Fitting data the main API (Application Programming Interface) implemented by scikitlearn is that of the estimator An estimator is any object that

from data it may be a classification regression or clustering algorithm or a transformer that extracts filters useful

features from raw data

All estimator objects expose a fit method that takes a dataset usually a 2d array

estimator.fit(data)

Estimator parameters All the parameters of an estimator can be set when it is instantiated or by modifying the

corresponding attribute

estimator.estimator_param1 param22

estimator.param1

1

Estimated parameters When data is fitted with an estimator parameters are estimated from the data at hand All the

estimated parameters are attributes of the estimator object ending by an underscore

estimator.estimated_param

158 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

222 Supervised learning predicting an output variable from highdimensional observations

The problem solved in supervised learning

Supervised learning consists in learning the link between two datasets the observed data X and an external variable y that we are trying to predict usually called “target” or “labels” Most often y is a 1D array of length n_{samples}

All supervised estimators in scikitlearn implement a `fit(X, y)` method to fit the model and a `predict(X)` method that given unlabeled observations X returns the predicted labels y

Vocabulary classification and regression

If the prediction task is to classify the observations in a set of finite labels in other words to “name” the objects observed the task is said to be a classification task On the other hand if the goal is to predict a continuous target variable it is said to be a regression task

When doing classification in scikitlearn y is a vector of integers or strings

Note See the Introduction to machine learning with scikitlearn Tutorial for a quick runthrough on the basic machine learning vocabulary used within scikitlearn

Nearest neighbor and the curse of dimensionality

Classifying irises

22 A tutorial on statistical learning for scientific data processing 159

scikitlearn user guide Release 0213

The iris dataset is a classification task consisting in identifying 3 different types of irises Setosa Versicolour and Virginica from their petal and sepal length and width

```
import numpy as np
from sklearn import datasets
iris = datasets.load_iris()
iris_X = iris.data
iris_y = iris.target
np.unique(iris_y)
array([0, 1, 2])
```

kNearest neighbors classifier

The simplest possible classifier is the nearest neighbor given a new observation X_{test} find in the training set ie the data used to train the estimator the observation with the closest feature vector Please see the Nearest Neighbors section of the online Scikitlearn documentation for more information about this type of classifier

Training set and testing set

While experimenting with any learning algorithm it is important not to test the prediction of an estimator on the data used to fit the estimator as this would not be evaluating the performance of the estimator on new data This is why datasets are often split into train and test data

scikitlearn user guide Release 0213

KNN k nearest neighbors classification example

Split iris data in train and test data

A random permutation to split the data randomly

`np.random.seed(0)`

`indices = np.random.permutation(len(irisX))`

`irisXtrain = irisX[indices[:10]]`

`irisYtrain = irisY[indices[:10]]`

`irisXtest = irisX[indices[10:]]`

`irisYtest = irisY[indices[10:]]`

Create and fit a nearest neighbor classifier

`from sklearn.neighbors import KNeighborsClassifier`

`knn = KNeighborsClassifier`

`knn.fit(irisXtrain, irisYtrain)`

`KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',`

`metric_params=None, n_jobs=None, n_neighbors=5, p=2,`

`weights='uniform')`

`knn.predict(irisXtest)`

`array([1, 2, 1, 0, 0, 0, 2, 1, 2, 0])`

`irisYtest`

`array([1, 1, 1, 0, 0, 0, 2, 1, 2, 0])`

The curse of dimensionality

For an estimator to be effective you need the distance between neighboring points to be less than some value ϵ which depends on the problem. In one dimension this requires on average $\sim 1/\epsilon$ points. In the context of the above ϵ -NN example if the data is described by just one feature with values ranging from 0 to 1 and with n training observations then new data will be no further away than $1/n$. Therefore the nearest neighbor decision rule will be efficient as soon as $1/n$ is small compared to the scale of between-class feature variations.

If the number of features is d you now require $\sim 1/\epsilon^d$ points. Let's say that we require 10 points in one dimension now 10^d points are required in d dimensions to pave the 0-1 space. As d becomes large the number of training points required for a good estimator grows exponentially.

For example if each point is just a single number 8 bytes then an effective ϵ -NN estimator in a paltry ~ 20 dimensions would require more training data than the current estimated size of the entire internet ~ 1000 Exabytes or 22 A tutorial on statistical learning for scientific data processing 161

scikitlearn user guide Release 0213

so

This is called the curse of dimensionality and is a core problem that machine learning addresses

Linear model from regression to sparsity

Diabetes dataset

The diabetes dataset consists of 10 physiological variables age sex weight blood pressure measure on 442 patients and an indication of disease progression after one year

diabetes datasetsloaddiabetes

diabetesXtrain diabetesdata20

diabetesXtest diabetesdata20

diabetesytrain diabetestarget20

diabetesytest diabetestarget20

The task at hand is to predict disease progression from physiological variables

Linear regression

LinearRegression in its simplest form fits a linear model to the data set by adjusting a set of parameters in order to make the sum of the squared residuals of the model as small as possible

Linear models

- data
- target variable
- Coefficients
- Observation noise

from sklearn import linearmodel

regr linearmodelLinearRegression

regrfitdiabetesXtrain diabetesytrain

LinearRegressioncopyXTrue fitinterceptTrue njobsNone

normalizeFalse

printregrcoef

030349955 23763931533 51053060544 32773698041 81413170937

49281458798 10284845219 18460648906 74351961675 7609517222

162 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213
The mean square error
npmeanregpredictdiabetesXtest diabetesytest 2

200456760268
Explained variance score 1 is perfect prediction
and 0 means that there is no linear relationship
between X and y
regrscorediabetesXtest diabetesytest

05850753022690
Shrinkage
If there are few data points per dimension noise in the observations induces high variance
X npc 5 1T
y 5 1
test npc 0 2T
regr linearmodelLinearRegression
import matplotlib.pyplot as plt
pltfigure
nprandomseed0
for inrange6
thisX 1 nprandomnormalsize2 1 X
regrfittthisX y
pltplottest regrpredicttest
pltscatterthisX y s3

A solution in highdimensional statistical learning is to shrink the regression coefficients to zero any
two randomly chosen set of observations are likely to be uncorrelated This is called Ridge regression
22 A tutorial on statistcallearning for scientific data processing 163

```
scikitlearn user guide Release 0213
regr linearmodelRidgealpha1
pltfigure
nprandomseed0
for inrange6
thisX 1 nprandomnormalsize2 1 X
regrfitthisX y
pltplottest regrpredicttest
pltscatterthisX y s3
```

This is an example of biasvariance tradeoff the larger the ridge alpha parameter the higher the bias and the lower the variance

We can choose alpha to minimize left out error this time using the diabetes dataset rather than our synthetic data

```
alphas nplogspace4 1 6
printregrsetparamsalphaalpha
fitdiabetesXtrain diabetesytrain
scorediabetesXtest diabetesytest
for alphainalphas
```

05851110683883 05852073015444 05854677540698
05855512036503 05830717085554 057058999437

Note Capturing in the fitted parameters noise that prevents the model to generalize to new data is called overfitting

The bias introduced by the ridge regression is called a regularization

Sparsity
Fitting only features 1 and 2

scikitlearn user guide Release 0213

Note A representation of the full diabetes dataset would involve 11 dimensions 10 feature dimensions and one of the target variable It is hard to develop an intuition on such representation but it may be useful to keep in mind that it would be a fairly empty space

We can see that although feature 2 has a strong coefficient on the full model it conveys little information on ywhen considered with feature 1

To improve the conditioning of the problem ie mitigating the The curse of dimensionality it would be interesting to select only the informative features and set noninformative ones like feature 2 to 0 Ridge regression will decrease their contribution but not set them to zero Another penalization approach called Lasso least absolute shrinkage and selection operator can set some coefficients to zero Such methods are called sparse method and sparsity can be seen as an application of Occam’s razor prefer simpler models

```
regr = linearmodelLasso
scores = regr.set_params(alpha=alpha)
fit_diabetes_X_train, diabetes_y_train =
score_diabetes_X_test, diabetes_y_test =
for alpha in alphas:
    best_alpha = alpha
    scores_index_max_scores =
    regr.alpha_ = best_alpha
    regr.fit(diabetes_X_train, diabetes_y_train)
```

```
Lasso(alpha=0.0025118864315095794, copy_X=True, fit_intercept=True,
max_iter=1000, normalize=False, positive=False, precompute=False,
random_state=None, selection_cyclic_tol=0.00001, warm_start=False)
print(regr.coef_)
0 21243764548 51719478111 31377959962 1608303982 0
18719554705 6938229038 50866011217 7184239008
22 A tutorial on statistical learning for scientific data processing 165
```

scikitlearn user guide Release 0213

Different algorithms for the same problem

Different algorithms can be used to solve the same mathematical problem For instance the Lasso object in scikit learn solves the lasso regression problem using a coordinate descent method that is efficient on large datasets However scikitlearn also provides the LassoLars object using the LARS algorithm which is very efficient for problems in which the weight vector estimated is very sparse ie problems with very few observations

Classification

For classification as in the labeling iris task linear regression is not the right approach as it will give too much weight to data far from the decision frontier A linear approach is to fit a sigmoid function or logistic function

$\sigma(\text{offset}) = \frac{1}{1 + \exp(-\text{offset})}$

$\log \text{linear model}$

`LogisticRegression(solver='lbfgs', C=1e5)`

`multiclassmultinomial`

`LogisticRegression`

`LogisticRegression(C=1000000, class_weight=None, dual=False,`

`fit_intercept=True, intercept_scaling=1, l1_ratio=None, max_iter=100,`

`multiclassmultinomial, n_jobs=None, penalty='l2', random_state=None,`

`solver='lbfgs', tol=0.0001, verbose=0, warm_start=False)`

This is known as LogisticRegression

166 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

Multiclass classification

If you have several classes to predict an option often used is to fit oneversusall classifiers and then use a voting heuristic for the final decision

Shrinkage and sparsity with logistic regression

TheCparameter controls the amount of regularization in the LogisticRegression object a large value

forCresults in less regularization penaltyl2 gives Shrinkage ie nonsparse coefficients while

penaltyl1 gives Sparsity

Exercise

Try classifying the digits dataset with nearest neighbors and a linear model Leave out the last 10 and test

prediction performance on these observations

from sklearn import datasets neighbors linearmodel

digits datasetsloaddigits

Xdigits digitsdata digitsdatamax

ydigits digitstarget

Solutionautoexamplesexerciseplotdigitsclassificationexercisepy

Support vector machines SVMs

Linear SVMs

Support Vector Machines belong to the discriminant model family they try to find a combination of samples to build

a plane maximizing the margin between the two classes Regularization is set by the Cparameter a small value for C

means the margin is calculated using many or all of the observations around the separating line more regularization

a large value for Cmeans the margin is calculated on observations close to the separating line less regularization

Unregularized SVM (Support Vector Machine) Regularized SVM (Support Vector Machine) default

22 A tutorial on statisticallearning for scientific data processing 167

scikitlearn user guide Release 0213

Example

•Plot different SVM (Support Vector Machine) classifiers in the iris dataset

SVMs can be used in regression – SVR (Support Vector Regression) Support Vector Regression– or in classification – SVC (Support Vector Classification)

from sklearn import svm

svc = svm.SVC(kernel='linear')

svc.fit(X_train, y_train)

SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,

decision_function_shape='ovr', degree=3, gamma='auto_deprecated',

kernel='linear', max_iter=1, probability=False, random_state=None,

shrinking=True, tol=0.001, verbose=False)

Warning: Normalizing data

For many estimators including the SVMs having datasets with unit standard deviation for each feature is important

to get good prediction

Using kernels

Classes are not always linearly separable in feature space. The solution is to build a decision function that is not linear

but may be polynomial instead. This is done using the kernel trick that can be seen as creating a decision energy by

positioning kernels on observations

Linear kernel Polynomial kernel

svc = svm.SVC(kernel='linear') svc = svm.SVC(kernel='poly',

degree=3,

degree polynomial degree)

168 Chapter 2 scikitlearn Tutorials

[scikitlearn user guide Release 0213](#)

[RBF \(Radial Basis Function\) kernel Radial Basis Function](#)

[svc svmSVCKernelrbf](#)

[gamma inverse of size of](#)

[radial kernel](#)

[Interactive example](#)

[See the SVM \(Support Vector Machine\) GUI to download svmguipy add data points of both classes with right and left button fit the model and change parameters and data](#)

[22 A tutorial on statisticallearning for scientific data processing 169](#)

scikitlearn user guide Release 0213

Exercise

Try classifying classes 1 and 2 from the iris dataset with SVMs with the 2 first features Leave out 10 of each class and test prediction performance on these observations

Warning the classes are ordered do not leave out the last 10 you would be testing on only one class

Hint You can use the decisionfunction method on a grid to get intuitions

iris datasetsloadiris

X irisdata

y iristarget

X Xy 0 2

y yy 0

Solutionautoexamplesexercisesplotirisexercisepy

223 Model selection choosing estimators and their parameters

Score and crossvalidated scores

As we have seen every estimator exposes a score method that can judge the quality of the fit or the prediction on

new data Bigger is better

from sklearn import datasets svm

digits datasetsloaddigits

Xdigits digitsdata

ydigits digitstarget

svc svmSVCC1 kernellinear

svcfitsvdigits100 ydigits100scoreXdigits100 ydigits100

098

To get a better measure of prediction accuracy which we can use as a proxy for goodness of fit of the model we can

successively split the data in folds that we use for training and testing

import numpy as np

Xfolds nparraysplitXdigits 3

yfolds nparraysplitydigits 3

scores list

for kinrange3

We use list to copy in order to pop later on

Xtrain listXfolds

Xtest Xtrainpopk

Xtrain npconcatenateXtrain

ytrain listyfolds

ytest ytrainpopk

ytrain npconcatenateytrain

scoresappendsvcfitsvdigits100 ytrainytrainscoreXtest ytest

printscores

0934 0956 0939

This is called a KFold crossvalidation

170 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

Crossvalidation generators

Scikitlearn has a collection of classes which can be used to generate lists of traintest indices for popular cross validation strategies

They expose a split method which accepts the input dataset to be split and yields the traintest set indices for each iteration of the chosen crossvalidation strategy

This example shows an example usage of the split method

```
from sklearn.model_selection import KFold crossvalscore
```

```
X = a a a b b c c c c c
```

```
kfold = KFold(n_splits=5)
```

```
for train_indices, test_indices in kfold.split(X):
```

```
    print('Train: %s, Test: %s' % (train_indices, test_indices))
```

```
Train: [2 3 4 5 6 7 8 9] Test: [0 1]
```

```
Train: [0 1 4 5 6 7 8 9] Test: [2 3]
```

```
Train: [0 1 2 3 6 7 8 9] Test: [4 5]
```

```
Train: [0 1 2 3 4 5 8 9] Test: [6 7]
```

```
Train: [0 1 2 3 4 5 6 7] Test: [8 9]
```

The crossvalidation can then be performed easily

```
svc.fit(X_train, y_train).score(X_test, y_test)
```

```
for train, test in kfold.split(X_digits):
```

```
    0.963 0.922 0.963 0.963 0.930
```

The crossvalidation score can be directly calculated using the crossvalscore helper Given an estimator the crossvalidation object and the input dataset the crossvalscore splits the data repeatedly into a training and a testing set trains the estimator using the training set and computes the scores based on the testing set for each iteration of crossvalidation

By default the estimator's score method is used to compute the individual scores

Refer the metrics module to learn more on the available scoring methods

```
cross_val_score(svc, X_digits, y_digits, cv=kfold, n_jobs=1)
```

```
array([0.96388889, 0.92222222, 0.9637883, 0.9637883, 0.93036212])
```

n_jobs=1 means that the computation will be dispatched on all the CPUs of the computer

Alternatively the scoring argument can be provided to specify an alternative scoring method

```
cross_val_score(svc, X_digits, y_digits, cv=kfold,
```

```
                scoring=precision_macro)
```

```
array([0.96578289, 0.92708922, 0.96681476, 0.96362897, 0.93192644])
```

Crossvalidation generators

KFold n_splits shuffle random

StratifiedKFold n_splits

shuffle random state GroupKFold n_splits

Splits it into K folds trains on K-1

and then tests on the left-out Same as KFold but preserves the

class distribution within each fold Ensures that the same group is not in

both testing and training sets

22 A tutorial on statistical learning for scientific data processing 171

scikitlearn user guide Release 0.21.3

ShuffleSplit n_splits
test_size train_size random_state
StratifiedShuffleSplit GroupShuffleSplit
Generates train/test indices based
on random permutation. Same as shuffle split but preserves the
class distribution within each iteration. Ensures that the same group is not
in both testing and training sets

LeaveOneGroupOut LeavePGroupsOut n_groups LeaveOneOut
Takes a group array to group observations. Leave P groups out. Leave one observation out

LeavePOut p PredefinedSplit
Leave P observations out. Generates train/test indices based on predefined splits

Exercise

On the
digits dataset plot the crossvalidation score of a SVC (Support Vector Classification) estimator with an linear kernel as a function of parameter C
use a logarithmic grid of points from 1 to 10

```
import numpy as np
from sklearn.model_selection import cross_val_score
from sklearn import datasets, svm
digits = datasets.load_digits()
X, y = digits.data, digits.target
svc = svm.SVC(kernel='linear')
Cs = np.logspace(0, 10, 10)
172 Chapter 2 scikitlearn Tutorials
```

scikitlearn user guide Release 0213

Solution Crossvalidation on Digits Dataset Exercise

Gridsearch and crossvalidated estimators

Gridsearch

scikitlearn provides an object that given data computes the score during the fit of an estimator on a parameter grid and chooses the parameters to maximize the crossvalidation score This object takes an estimator during the construction and exposes an estimator API (Application Programming Interface)

from sklearnmodelselection import GridSearchCV crossvalscore

Cs nplogspace6 1 10

clf GridSearchCVestimatorsvc paramgriddictCCs

njobs1

clffitXdigits1000 ydigits1000

GridSearchCVcvNone

clfbestscore

0925

clfbestestimatorC

00077

Prediction performance on test set is not as good as on train set

clfscoreXdigits1000 ydigits1000

0943

By default the GridSearchCV uses a 3fold crossvalidation However if it detects that a classifier is passed rather than a regressor it uses a stratified 3fold The default will change to a 5fold crossvalidation in version 022

Nested crossvalidation

crossvalscoreclf Xdigits ydigits

array0938 0963 0944

Two crossvalidation loops are performed in parallel one by the GridSearchCV estimator to set gamma and the other one by crossvalscore to measure the prediction performance of the estimator The resulting scores are unbiased estimates of the prediction score on new data

Warning You cannot nest objects with parallel computing njobs different than 1

Crossvalidated estimators

Crossvalidation to set a parameter can be done more efficiently on an algorithmbyalgorithm basis This is why for certain estimators scikitlearn exposes Crossvalidation evaluating estimator performance estimators that set their parameter automatically by crossvalidation

from sklearn import linearmodel datasets

lasso linearmodelLassoCVcv3

diabetes datasetsloaddiabetes

Xdiabetes diabetesdata

ydiabetes diabetestarget

22 A tutorial on statisticallearning for scientific data processing 173

scikitlearn user guide Release 0213

lassofitXdiabetes ydiabetes

LassoCValphasNone copyXTrue cv3 eps0001 fitinterceptTrue

maxiter1000 nalphas100 njobsNone normalizeFalse

positiveFalse precomputeauto randomstateNone

selectioncyclic tol00001 verboseFalse

The estimator chose automatically its lambda

lassoalpha

001229

These estimators are called similarly to their counterparts with 'CV (Cross-Validation)' appended to their name

Exercise

On the diabetes dataset find the optimal regularization parameter alpha

Bonus How much can you trust the selection of alpha

from sklearn import datasets

from sklearnlinearmodel import LassoCV

from sklearnlinearmodel import Lasso

from sklearnmodelselection import KFold

from sklearnmodelselection import GridSearchCV

diabetes datasetsloadaddiabetes

X diabetesdata150

Solution Crossvalidation on diabetes Dataset Exercise

224 Unsupervised learning seeking representations of the data

Clustering grouping observations together

The problem solved in clustering

Given the iris dataset if we knew that there were 3 types of iris but did not have access to a taxonomist to label

them we could try a clustering task split the observations into wellseparated group called clusters

174 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

Kmeans clustering

Note that there exist a lot of different clustering criteria and associated algorithms The simplest clustering algorithm is Kmeans

from sklearn import cluster datasets

iris datasetsloadiris

Xiris irisdata

yiris iristarget

kmeans clusterKMeansnclusters3

kmeansfitXiris

KMeansalgorithmauto copyxTrue initkmeans

printkmeanslabels10

1 1 1 1 1 0 0 0 0 2 2 2 2 2

printyiris10

0 0 0 0 0 1 1 1 1 2 2 2 2 2

Warning There is absolutely no guarantee of recovering a ground truth First choosing the right number of clusters is hard Second the algorithm is sensitive to initialization and can fall into local minima although scikit

learn employs several tricks to mitigate this issue

Bad initialization 8 clusters Ground truth

Don't overinterpret clustering results

Application example vector quantization

Clustering in general and KMeans in particular can be seen as a way of choosing a small number of exemplars to compress the information The problem is sometimes known as vector quantization For instance this can be used

22 A tutorial on statisticallearning for scientific data processing 175

scikitlearn user guide Release 0213

to posterize an image

```
import scipy as sp
```

```
try
```

```
face = sp.facegray True
```

```
except AttributeError
```

```
from scipy import misc
```

```
face = misc.facegray True
```

```
X = face.reshape(1, 1) # We need an nsample nfeature array
```

```
kmeans = cluster.KMeans(nclusters=5, ninit=1)
```

```
kmeans.fit(X)
```

```
KMeans.algorthmauto = copyx=True, initkmeans
```

```
values = kmeans.cluster_centers_.squeeze
```

```
labels = kmeans.labels
```

```
facecompressed = np.choose(labels, values)
```

```
facecompressed.shape = face.shape
```

Raw image Kmeans quantization Equal bins Image histogram

Hierarchical agglomerative clustering Ward

A Hierarchical clustering method is a type of cluster analysis that aims to build a hierarchy of clusters. In general, the

various approaches of this technique are either

- Agglomerative (bottomup) approaches: each observation starts in its own cluster, and clusters are iteratively merged in such a way to minimize a linkage criterion. This approach is particularly interesting when the clusters of interest are made of only a few observations. When the number of clusters is large, it is much more computationally efficient than kmeans.

- Divisive (topdown) approaches: all observations start in one cluster, which is iteratively split as one moves down the hierarchy. For estimating large numbers of clusters, this approach is both slow (due to all observations starting as one cluster) and statistically illposed.

Connectivity-constrained clustering

With agglomerative clustering, it is possible to specify which samples can be clustered together by giving a connectivity graph. Graphs in scikitlearn are represented by their adjacency matrix. Often, a sparse matrix is used. This

can be useful for instance to retrieve connected regions, sometimes also referred to as connected components. When

176 Chapter 2 scikitlearn Tutorials

```
scikitlearn user guide Release 0.21.3
clustering an image
from scipy.ndimage.filters import gaussian_filter
import matplotlib.pyplot as plt
import skimage
from skimage.data import coins
from skimage.transform import rescale
from sklearn.feature_extraction.image import grid_to_graph
from sklearn.cluster import AgglomerativeClustering
# these were introduced in skimage 0.14
if LooseVersion(skimage.__version__) < LooseVersion('0.14'):
    rescale_params = {'antialiasing': False, 'multichannel': False}
else:
    rescale_params = {}
```

```
# Generate data
orig_coins = coins
# Resize it to 20% of the original size to speed up the processing
# Applying a Gaussian filter for smoothing prior to downscaling
# reduces aliasing artifacts
smoothed_coins = gaussian_filter(orig_coins, sigma=2)
```

Feature agglomeration

We have seen that sparsity could be used to mitigate the curse of dimensionality i.e. an insufficient amount of observations compared to the number of features. Another approach is to merge together similar features: feature agglomeration. This approach can be implemented by clustering in the feature direction, in other words clustering.

22 A tutorial on statistical learning for scientific data processing 177

scikitlearn user guide Release 0213

the transposed data

digits datasetsloaddigits

images digitsimages

X npreshapeimages lenimages 1

connectivity gridtograph images0shape

agglo clusterFeatureAgglomerationconnectivityconnectivity

 nclusters32

agglofitX

FeatureAgglomerationaffinityeuclidean computefulltreeauto

Xreduced agglotransformX

Xapprox aggloinversettransformXreduced

imagesapprox npreshapeXapprox imagesshape

transform andinversettransform methods

Some estimators expose a transform method for instance to reduce the dimensionality of the dataset

Decompositions from a signal to components and loadings

Components and loadings

If X is our multivariate data then the problem that we are trying to solve is to rewrite it on a different observational basis we want to learn loadings L and a set of components C such that $X = L C$ Different criteria exist to choose the components

Principal component analysis PCA (Principal Component Analysis)

Principal component analysis PCA (Principal Component Analysis) selects the successive components that explain the maximum variance in the signal

178 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

The point cloud spanned by the observations above is very flat in one direction one of the three univariate features can almost be exactly computed using the other two PCA (Principal Component Analysis) finds the directions in which the data is not flat When used to transform data PCA (Principal Component Analysis) can reduce the dimensionality of the data by projecting on a principal subspace

```
Create a signal with only 2 useful dimensions
x1 = np.random.randn(100)
x2 = np.random.randn(100)
x3 = x1 * x2
X = np.column_stack((x1, x2, x3))

from sklearn import decomposition
pca = decomposition.PCA()
pca.fit(X)
PCAcopy=True iteratedpowerauto ncomponents=None randomstate=None
svdsolverauto tol00 whitenFalse
print(pca.explained_variance_)
218565811e00 119346747e00 843026679e32

As we can see only the 2 first components are useful
pca.n_components_
2
X_reduced = pca.fit_transform(X)
X_reduced.shape
(100, 2)
```

Independent Component Analysis ICA

Independent component analysis ICA selects components so that the distribution of their loadings carries a maximum amount of independent information It is able to recover nonGaussian independent signals

22 A tutorial on statistical learning for scientific data processing 179

```
scikitlearn user guide Release 0213
Generate sample data
import numpy as np
from scipy import signal
time = nplinspace0 10 2000
s1 = npsin2 time Signal 1 sinusoidal signal
s2 = npsignnpsin3 time Signal 2 square signal
s3 = signalsawtooth2 nppitime Signal 3 saw tooth signal
S = npcs1 s2 s3
S = 02 nprandomnormalsizeSshape Add noise
S = Sstdaxis0 Standardize data
Mix data
A = nparray1 1 1 05 2 1 15 1 2 Mixing matrix
X = npdotS AT Generate observations
Compute ICA
ica = decompositionFastICA
S = icafittransformX Get the estimated sources
A = icamixingT
npallcloseX npdotS A icamean
True
225 Putting it all together
180 Chapter 2 scikitlearn Tutorials
```

scikitlearn user guide Release 0213

Pipelining

We have seen that some estimators can transform data and that some estimators can predict variables We can also create combined estimators

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import datasets
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.linear_model import SGDClassifier
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
Define a pipeline to search for the best combination of PCA (Principal Component Analysis) truncation
and classifier regularization
logistic SGDClassifier loss=log penalty=l2 earlystopping=True
maxiter=10000 tol=1e-5 randomstate=0
pca PCA (Principal Component Analysis)
pipe Pipeline(steps=[pca, logistic])
digits = datasets.load_digits()
Xdigits, ydigits = digits.data, digits.target
Parameters of pipelines can be set using “” separated parameter names
paramgrid = {
    'pca__components': [5, 20, 30, 40, 50, 64],
    'logistic__alpha': np.logspace(-4, 4, 5)
}
```

```
search = GridSearchCV(pipe, paramgrid, iid=False, cv=5)
search.fit(Xdigits, ydigits)
22 A tutorial on statistical learning for scientific data processing 181
```

```

scikitlearn user guide Release 0213
printBest parameter CV (Cross-Validation) score 03f searchbestscore
printsearchbestparams
Plot the PCA (Principal Component Analysis) spectrum
pcafitXdigits
fig ax0 ax1 pltsubplotsnrows2 sharex True figsize6 6
ax0plotpcaexplainedvarianceratio linewidth2
ax0setylabelPCA explained variance
ax0axvlinsearchbestestimatornamedstepspcancomponents
linestyle labelIncomponents chosen
Face recognition with eigenfaces
The dataset used in this example is a preprocessed excerpt of the “Labeled Faces in the Wild” also known as LFW
httpviswwwwsumasssedulfwlwfwnneledtgz 233MB
```

Faces recognition example using eigenfaces and SVMs

The dataset used in this example is a preprocessed excerpt of the Labeled Faces in the Wild aka LFW
httpviswwwwsumasssedulfwlwfwnneledtgz 233MB
LFW httpviswwwwsumasssedulfw
Expected results for the top 5 most represented people in the dataset

precision recall f1score support

| | | | | |
|-------------------|-----|-----|-----|-----|
| Ariel Sharon | 067 | 092 | 077 | 13 |
| Colin Powell | 075 | 078 | 076 | 60 |
| Donald Rumsfeld | 078 | 067 | 072 | 27 |
| George W Bush | 086 | 086 | 086 | 146 |
| Gerhard Schroeder | 076 | 076 | 076 | 25 |
| Hugo Chavez | 067 | 067 | 067 | 15 |
| Tony Blair | 081 | 069 | 075 | 36 |
| avg total | 080 | 080 | 080 | 322 |

```

from time import time
import logging
import matplotlib.pyplot as plt
from sklearnmodelselection import traintestsplit
from sklearnmodelselection import GridSearchCV
from sklearndatasets import fetchlfwpeople
from sklearnmetrics import classificationreport
from sklearnmetrics import confusionmatrix
182 Chapter 2 scikitlearn Tutorials
```

```
scikitlearn user guide Release 0213
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.svm import SVC (Support Vector Classification)
printdoc
    Display progress logs on stdout
logging.basicConfig(level=logging.INFO, format='%(asctime)s %(message)s')
```

```
Download the data if not already on disk and load it as numpy arrays
lfwpeople = fetch_lfw_people(min_faces_per_person=70, resize=0.4)
introspect the images arrays to find the shapes for plotting
n_samples, h, w = lfwpeople.images.shape
for machine learning we use the 2 data directly as relative pixel
positions info is ignored by this model
X = lfwpeople.data
n_features = X.shape[1]
the label to predict is the id of the person
y = lfwpeople.target
target_names = lfwpeople.target_names
n_classes = target_names.shape[0]
print('Total dataset size: %d' % n_samples)
print('n_samples: %d' % n_samples)
print('n_features: %d' % n_features)
print('n_classes: %d' % n_classes)
```

```
Split into a training set and a test set using a stratified k fold
split into a training and testing set
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=42)
```

```
Compute a PCA (Principal Component Analysis) eigenfaces on the face dataset treated as unlabeled
dataset unsupervised feature extraction - dimensionality reduction
n_components = 150
print('Extracting the top %d eigenfaces from %d faces' % (n_components, X_train.shape[0]))
t0 = time
pca = PCA(n_components=n_components, svd_solver='randomized',
          whiten=True).fit(X_train)
print('done in %.3fs' % (time - t0))
eigenfaces = pca.components_.reshape((n_components, h, w))
print('Projecting the input data on the eigenfaces orthonormal basis')
t0 = time
22 A tutorial on statistical learning for scientific data processing 183
```

scikitlearn user guide Release 0213

Xtrainpca pcatransformXtrain
Xtestpca pcatransformXtest
printdone in 03fs time t0

 Train a SVM (Support Vector Machine) classification model
printFitting the classifier to the training set
t0 time
paramgrid C 1e3 5e3 1e4 5e4 1e5
gamma 00001 00005 0001 0005 001 01
clf GridSearchCVSVCKernelrbf classweightbalanced
paramgrid cv5 iid False
clf clffitXtrainpca ytrain
printdone in 03fs time t0
printBest estimator found by grid search
printclfbestestimator

 Quantitative evaluation of the model quality on the test set
printPredicting peoples names on the test set
t0 time
ypred clfpredictXtestpca
printdone in 03fs time t0
printclassificationreportytest ypred targetnamestargetnames
printconfusionmatrixytest ypred labelsrangenclasses

 Qualitative evaluation of the predictions using matplotlib
defplotgalleryimages titles h w nrow3 ncol4
 Helper function to plot a gallery of portraits
pltfigurefigsize18 ncol 24 nrow
pltsubplotsadjustbottom0 left01 right99 top90 hspace35
foriinrangencol ncol
 pltsubplotnrow ncol i 1
 pltimshowimagesiresshapeh w cmappltcmgray
 plttitletitlesi size12
 pltxticks
 pltyticks
 plot the result of the prediction on a portion of the test set
 deftitleypred ytest targetnames i
 predname targetnamesypredirsplit 11
 truename targetnamesytestirsplit 11
 returnpredicted sntrue s predname truename
 predictiontitles titleypred ytest targetnames i
 foriinrangeypredshape0
 plotgalleryXtest predictiontitles h w

184 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

plot the gallery of the most significant eigenfaces

eigenfacetitles eigenface d iforiinrangeeigenfacesshape0

plotgalleryeigenfaces eigenfacetitles h w

pltshow

Prediction Eigenfaces

Expected results for the top 5 most represented people in the dataset

precision recall f1score support

GerhardSchroeder 091 075 082 28

DonaldRumsfeld 084 082 083 33

TonyBlair 065 082 073 34

ColinPowell 078 088 083 58

GeorgeWBush 093 086 090 129

avg total 086 084 085 282

Open problem Stock Market Structure

Can we predict the variation in stock prices for Google over a given time frame

Learning a graph structure

226 Finding help

The project mailing list

If you encounter a bug with scikitlearn or something that needs clarification in the docstring or the online documentation please feel free to ask on the Mailing List

22 A tutorial on statisticallearning for scientific data processing 185

scikitlearn user guide Release 0213

QA communities with Machine Learning practitioners

Quoracom Quora has a topic for Machine Learning related questions that also features some interesting discussions <httpswwwquoracomtopicMachineLearning>

Stack Exchange The Stack Exchange family of sites hosts multiple subdomains for Machine Learning questions

- 'An excellent free online course for Machine Learning taught by Professor Andrew Ng of Stanford' <httpswwwcourseraorglearnmachinelearning>

- 'Another excellent free online course that takes a more general approach to Artificial Intelligence' <httpswwwudacitycomcourseintrotoartificialintelligence-cs271>

23 Working With Text Data

The goal of this guide is to explore some of the main scikitlearn tools on a single practical task analyzing a collection of text documents newsgroups posts on twenty different topics

In this section we will see how to

- load the file contents and the categories
- extract feature vectors suitable for machine learning
- train a linear model to perform categorization
- use a grid search strategy to find a good configuration of both the feature extraction components and the classifier

231 Tutorial setup

To get started with this tutorial you must first install scikitlearn and all of its required dependencies

Please refer to the installation instructions page for more information and for systemspecific instructions

The source of this tutorial can be found within your scikitlearn folder

`scikitlearndoctutorialtextanalytics`

The source can also be found on Github

The tutorial folder should contain the following subfolders

- `rst` files the source of the tutorial document written with sphinx
- `data` folder to put the datasets used during the tutorial
- `skeletons` sample incomplete scripts for the exercises
- `solutions` solutions of the exercises

You can already copy the skeletons into a new folder somewhere on your harddrive named

`sklearnututworkspace` where you will edit your own files for the exercises while keeping the original skeletons intact

`cp r skeletons workdirectorysklearnututworkspace`

186 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

Machine learning algorithms need data Go to each TUTORIALHOMEdata subfolder and run the fetchdatapy script from there after having read them first

For instance

```
cd TUTORIALHOMEdatalanguages
less fetchdatapy
python fetchdatapy
```

232 Loading the 20 newsgroups dataset

The dataset is called “Twenty Newsgroups” Here is the official description quoted from the website

The 20 Newsgroups data set is a collection of approximately 20000 newsgroup documents partitioned nearly evenly across 20 different newsgroups To the best of our knowledge it was originally collected by Ken Lang probably for his paper “Newsweeder Learning to filter netnews” though he does not explicitly mention this collection The 20 newsgroups collection has become a popular data set for experiments in text applications of machine learning techniques such as text classification and text clustering

In the following we will use the builtin dataset loader for 20 newsgroups from scikitlearn Alternatively it is possible to download the dataset manually from the website and use the sklearndatasetsloadfiles function by pointing it to the 20newsbydatetrain subfolder of the uncompressed archive folder

In order to get faster execution times for this first example we will work on a partial dataset with only 4 categories out of the 20 available in the dataset

```
categories altatheism socreligionchristian
compgraphics scimed
```

We can now load the list of files matching those categories as follows

```
from sklearndatasets import fetch20newsgroups
twentytrain fetch20newsgroupssubsettrain
categoriescategories shuffle True randomstate42
```

The returned dataset is a scikitlearn “bunch” a simple holder object with fields that can be both accessed as pythondict keys or object attributes for convenience for instance the targetnames holds the list of the requested category names

```
twentytraintargetnames
altatheism compgraphics scimed socreligionchristian
```

The files themselves are loaded in memory in the data attribute For reference the filenames are also available

```
twentytraindata
twentytrainfilenames
```

Let’s print the first lines of the first loaded file

```
printjointtwentytraindata0split n3
From sd345cityacuk Michael Collier
Subject Converting images to HP LaserJet III
NntpPostingHost hampton
printtwentytraintargetnamesttwentytraintarget0
compgraphics
```

scikitlearn user guide Release 0213

Supervised learning algorithms will require a category label for each document in the training set In this case the category is the name of the newsgroup which also happens to be the name of the folder holding the individual documents For speed and space efficiency reasons scikitlearn loads the target attribute as an array of integers that corresponds to the index of the category name in the targetnames list The category integer id of each sample is stored in the target attribute

twentytraintarget10

array1 1 3 3 3 3 3 2 2 2

It is possible to get back the category names as follows

for tinttwentytraintarget10

printtwentytraintargetnamest

compgraphics

compgraphics

socreligionchristian

socreligionchristian

socreligionchristian

socreligionchristian

socreligionchristian

scimed

scimed

scimed

You might have noticed that the samples were shuffled randomly when we called fetch20newsgroups

shuffleTrue randomstate42 this is useful if you wish to select only a subset of samples to quickly

train a model and get a first idea of the results before retraining on the complete dataset later

233 Extracting features from text files

In order to perform machine learning on text documents we first need to turn the text content into numerical feature vectors

Bags of words

The most intuitive way to do so is to use a bags of words representation

1 Assign a fixed integer id to each word occurring in any document of the training set for instance by building a dictionary from words to integer indices

2 For each document i count the number of occurrences of each word wand store it in X_{ij} as the value

of featurejwherejis the index of word win the dictionary

The bags of words representation implies that nfeatures is the number of distinct words in the corpus this number is typically larger than 100000

Ifnsamples 10000 storingXas a NumPy array of type float32 would require 10000 x 100000 x 4 bytes

4GB in RAM which is barely manageable on today's computers

Fortunately most values in X will be zeros since for a given document less than a few thousand distinct words will be used For this reason we say that bags of words are typically highdimensional sparse datasets We can save a lot of memory by only storing the nonzero parts of the feature vectors in memory

scikitlearn user guide Release 0213

scipy sparse matrices are data structures that do exactly this and scikitlearn has builtin support for these structures

Tokenizing text with scikitlearn

Text preprocessing tokenizing and filtering of stopwords are all included in CountVectorizer which builds a dictionary of features and transforms documents to feature vectors

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
countvect = CountVectorizer
```

```
X_train_counts = countvect.fit_transform(train_data)
```

```
X_train_counts.shape
```

```
(2257, 35788)
```

CountVectorizer supports counts of Ngrams of words or consecutive characters. Once fitted the vectorizer has built a dictionary of feature indices

```
countvect.vocabulary.get_algorithm
```

```
4690
```

The index value of a word in the vocabulary is linked to its frequency in the whole training corpus

From occurrences to frequencies

Occurrence count is a good start but there is an issue: longer documents will have higher average count values than shorter documents even though they might talk about the same topics

To avoid these potential discrepancies it suffices to divide the number of occurrences of each word in a document by the total number of words in the document. These new features are called tf for Term Frequencies

Another refinement on top of tf is to downscale weights for words that occur in many documents in the corpus and are therefore less informative than those that occur only in a smaller portion of the corpus

This downscaling is called tf-idf for "Term Frequency times Inverse Document Frequency"

Both tf and tf-idf can be computed as follows using TfidfTransformer

```
from sklearn.feature_extraction.text import TfidfTransformer
```

```
tf_transformer = TfidfTransformer(use_idf=False).fit(X_train_counts)
```

```
X_train_tf = tf_transformer.transform(X_train_counts)
```

```
X_train_tf.shape
```

```
(2257, 35788)
```

In the above example code we firstly use the fit method to fit our estimator to the data and secondly the

transform method to transform our count matrix to a tfidf representation. These two steps can be com

bined to achieve the same end result faster by skipping redundant processing. This is done through using the

fit_transform method as shown below and as mentioned in the note in the previous section

```
tfidf_transformer = TfidfTransformer
```

```
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)
```

```
X_train_tfidf.shape
```

```
(2257, 35788)
```

23 Working With Text Data 189

scikitlearn user guide Release 0213

234 Training a classifier

Now that we have our features we can train a classifier to try to predict the category of a post Let’s start with a naïve Bayes classifier which provides a nice baseline for this task scikitlearn includes several variants of this classifier the one most suitable for word counts is the multinomial variant

```
from sklearnnaivebayes import MultinomialNB
clf = MultinomialNBfitXtraintfidf twentytraintarget
```

To try to predict the outcome on a new document we need to extract the features using almost the same feature extract ing chain as before The difference is that we call transform instead offittransform on the transformers

since they have already been fit to the training set

docsnew God is love OpenGL on the GPU is fast

Xnewcounts countvecttransformdocsnew

Xnewtfidf tfidftransformertransformXnewcounts

predicted = clfpredictXnewtfidf

for doc category in zipdocsnew predicted

printrs doc twentytraintargetnamescategory

God is love socreligionchristian

OpenGL on the GPU is fast compgraphics

235 Building a pipeline

In order to make the vectorizer transformer classifier easier to work with scikitlearn provides a Pipeline class that behaves like a compound classifier

from sklearnpipeline import Pipeline

textclf = Pipeline

vect = CountVectorizer

tfidf = TfidfTransformer

clf = MultinomialNB

The names vect tfidf andclf classifier are arbitrary We will use them to perform grid search for suitable hyperparameters below We can now train the model with a single command

textclffittwentytraindata twentytraintarget

Pipeline

236 Evaluation of the performance on the test set

Evaluating the predictive accuracy of the model is equally easy

import numpy as np

twentytest = fetch20newsgroupssubsettest

categoriescategories shuffle True randomstate42

docstest = twentytestdata

predicted = textclfpredictdocstest

npmeanpredicted = twentytesttarget

08348

190 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

We achieved 835 accuracy Let’s see if we can do better with a linear support vector machine SVM (Support Vector Machine) which is widely regarded as one of the best text classification algorithms although it’s also a bit slower than naïve Bayes We can change the learner by simply plugging a different classifier object into our pipeline

```
from sklearn.linear_model import SGDClassifier
textclf = Pipeline(
    vect=CountVectorizer(),
    tfidf=TfidfTransformer(),
    clf=SGDClassifier(loss='hinge', penalty='l2',
        alpha=1e-3, random_state=42,
        max_iter=5, tol=None)
```

```
textclf.fit(train_data, train_target)
predicted = textclf.predict(doc_test)
np.mean(predicted == test_target)
0.9101
```

We achieved 913 accuracy using the SVM (Support Vector Machine) scikitlearn provides further utilities for more detailed performance analysis of the results

```
from sklearn import metrics
print(metrics.classification_report(test_target, predicted,
    target_names=test_target_names))
```

```
precision recall f1-score support
altatheism 0.95 0.80 0.87 319
compgraphics 0.87 0.98 0.92 389
scimed 0.94 0.89 0.91 396
socreligionchristian 0.90 0.95 0.93 398
accuracy 0.91 1.00 0.95 1502
macro avg 0.91 0.91 0.91 1502
weighted avg 0.91 0.91 0.91 1502
metrics.confusion_matrix(test_target, predicted)
array([[256, 11, 16, 36],
       [ 4, 380, 3, 2],
       [ 5, 35, 353, 3],
       [ 5, 11, 4, 378])
```

As expected the confusion matrix shows that posts from the newsgroups on atheism and Christianity are more often confused for one another than with computer graphics

237 Parameter tuning using grid search

We’ve already encountered some parameters such as use_idf in the TfidfTransformer. Classifiers tend to have many parameters as well eg MultinomialNB includes a smoothing parameter alpha and SGDClassifier has a penalty parameter alpha and configurable loss and penalty terms in the objective function see the module documentation or use the Python help function to get a description of these

Instead of tweaking the parameters of the various components of the chain it is possible to run an exhaustive search of the best parameters on a grid of possible values We try out all classifiers on either words or bigrams with or without idf and with a penalty parameter of either 0.01 or 0.001 for the linear SVM (Support Vector Machine)

```
scikitlearn user guide Release 0213
from sklearnmodelselection import GridSearchCV
parameters
vectngramrange 1 1 1 2
tfidfuseidf TrueFalse
clfalpha 1e2 1e3
```

Obviously such an exhaustive search can be expensive If we have multiple CPU cores at our disposal we can tell the grid searcher to try these eight parameter combinations in parallel with the njobs parameter If we give this parameter a value of 1 grid search will detect how many cores are installed and use them all

```
gsclf GridSearchCVtextclf parameters cv5 iid False njobs1
The grid search instance behaves like a normal scikitlearn model Let’s perform the search on a smaller subset
of the training data to speed up the computation
gsclf gsclffittwentytraindata400 twentytraintarget400
The result of calling fit on aGridSearchCV object is a classifier that we can use to predict
twentytraintargetnamesgsclfpredictGod is love0
socreligionchristian
```

```
The object’s bestscore andbestparams attributes store the best mean score and the parameters setting
corresponding to that score
gsclfbestscore
09
for paramname insortedparameterskeys
printsr paramname gsclfbestparamsparamname
```

```
clfalpha 0001
tfidfuseidf True
vectngramrange 1 1
```

A more detailed summary of the search is available at gsclfcvresults
Thecvresults parameter can be easily imported into pandas as a DataFrame for further inspection
Exercises

```
To do the exercises copy the content of the ‘skeletons’ folder as a new folder named ‘workspace’
cp r skeletons workspace
You can then edit the content of the workspace without fear of losing the original exercise instructions
Then fire an ipython shell and run the workinprogress script with
1 run workspaceexerciseXXscriptpy arg1 arg2 arg3
If an exception is triggered use debug to fireup a post mortem ipdb session
Refine the implementation and iterate until the exercise is solved
For each exercise the skeleton file provides all the necessary import statements boilerplate code to load the
data and sample code to evaluate the predictive accuracy of the model
```

scikitlearn user guide Release 0213

238 Exercise 1 Language identification

- Write a text classification pipeline using a custom preprocessor and CharNGramAnalyzer using data from Wikipedia articles as training set
- Evaluate the performance on some held out test set

ipython command line

```
run workspaceexercise01languagegetrainmodelpy datalanguagesparagraphs
```

239 Exercise 2 Sentiment Analysis on movie reviews

- Write a text classification pipeline to classify movie reviews as either positive or negative
- Find a good set of parameters using grid search
- Evaluate the performance on a held out test set

ipython command line

```
run workspaceexercise02sentimentpy datamoviereviewstxtsentoken
```

2310 Exercise 3 CLI text classification utility

Using the results of the previous exercises and the cPickle module of the standard library write a command line utility that detects the language of some text provided on stdin and estimate the polarity positive or negative if the text is written in English

Bonus point if the utility is able to give a confidence level for its predictions

2311 Where to from here

Here are a few suggestions to help further your scikitlearn intuition upon the completion of this tutorial

- Try playing around with the analyzer and token normalisation under CountVectorizer
- If you don't have labels try using Clustering on your problem
- If you have multiple labels per document eg categories have a look at the Multiclass and multilabel section
- Try using Truncated SVD for latent semantic analysis
- Have a look at using Outofcore Classification to learn from data that would not fit into the computer main memory
- Have a look at the Hashing Vectorizer as a memory efficient alternative to CountVectorizer

24 Choosing the right estimator

Often the hardest part of solving a machine learning problem can be finding the right estimator for the job

Different estimators are better suited for different types of data and different problems

The flowchart below is designed to give users a bit of a rough guide on how to approach problems with regard to which estimators to try on your data

24 Choosing the right estimator 193

scikitlearn user guide Release 0213

Click on any estimator in the chart below to see its documentation

25 External Resources Videos and Talks

For written tutorials see the Tutorial section of the documentation

251 New to Scientific Python

For those that are still new to the scientific Python ecosystem we highly recommend the Python Scientific Lecture

Notes This will help you find your footing a bit and will definitely improve your scikitlearn experience A basic

understanding of NumPy arrays is recommended to make the most of scikitlearn

252 External Tutorials

There are several online tutorials available which are geared toward specific subject areas

- Machine Learning for NeuroImaging in Python
- Machine Learning for Astronomical Data Analysis

253 Videos

- An introduction to scikitlearn Part I and Part II at Scipy 2013 by Gael Varoquaux Jake Vanderplas and Olivier

Grisel Notebooks on github

- Introduction to scikitlearn by Gael Varoquaux at ICML 2010

A three minute video from a very early stage of scikitlearn explaining the basic idea and approach

we are following

- Introduction to statistical learning with scikitlearn by Gael Varoquaux at SciPy 2011

An extensive tutorial consisting of four sessions of one hour The tutorial covers the basics of ma

chine learning many algorithms and how to apply them using scikitlearn The material correspond

ing is now in the scikitlearn documentation section A tutorial on statistcallearning for scientific

data processing

- Statistical Learning for Text Classification with scikitlearn and NLTK and slides by Olivier Grisel at PyCon

2011

Thirty minute introduction to text classification Explains how to use NLTK and scikitlearn to solve

realworld text classification tasks and compares against cloudbased solutions

- Introduction to Interactive Predictive Analytics in Python with scikitlearn by Olivier Grisel at PyCon 2012

3hours long introduction to prediction tasks using scikitlearn

- scikitlearn Machine Learning in Python by Jake Vanderplas at the 2012 PyData workshop at Google

Interactive demonstration of some scikitlearn features 75 minutes

- scikitlearn tutorial by Jake Vanderplas at PyData NYC 2012

Presentation using the online tutorial 45 minutes

194 Chapter 2 scikitlearn Tutorials

scikitlearn user guide Release 0213

Note Doctest Mode

The codeexamples in the above tutorials are written in a pythonconsole format If you wish to easily execute these examples in IPython use

doctestmode

in the IPythonconsole You can then simply copy and paste the examples directly into IPython without having to worry about removing the manually

25 External Resources Videos and Talks 195

CHAPTER

THREE

USER GUIDE

31 Supervised learning

311 Generalized Linear Models

The following are a set of methods intended for regression in which the target value is expected to be a linear combination of the features In mathematical notation if \hat{y} is the predicted value

$\hat{y} = \beta_0 + \beta_1 x_1 + \dots + \beta_n x_n$

Across the module we designate the vector $\beta = [\beta_0, \beta_1, \dots, \beta_n]$ as `coef` and β_0 as `intercept`

To perform classification with generalized linear models see Logistic regression

Ordinary Least Squares

`LinearRegression` fits a linear model with coefficients $\beta_0, \beta_1, \dots, \beta_n$ to minimize the residual sum of squares between the observed targets in the dataset and the targets predicted by the linear approximation Mathematically it solves a problem of the form

$$\min_{\beta} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

`LinearRegression` will take in its `fit` method arrays `X` `y` and will store the coefficients β of the linear model in its `coef` member

```
scikitlearn user guide Release 0.21.3
from sklearn import linear_model
reg = linear_model.LinearRegression()
reg.fit(X, y)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
normalize=False,
reg_coef=0.05)
```

The coefficient estimates for Ordinary Least Squares rely on the independence of the features. When features are correlated and the columns of the design matrix X have an approximate linear dependence, the design matrix becomes close to singular and as a result the least squares estimate becomes highly sensitive to random errors in the observed target, producing a large variance. This situation of multicollinearity can arise, for example, when data are collected without an experimental design.

- Examples
- Linear Regression Example
- Ordinary Least Squares Complexity

The least squares solution is computed using the singular value decomposition of X . If X is a matrix of shape $n_{\text{samples}} \times n_{\text{features}}$, this method has a cost of $O(n_{\text{samples}}^2 n_{\text{features}})$ assuming that $n_{\text{samples}} \geq n_{\text{features}}$.

Ridge Regression

Ridge regression addresses some of the problems of Ordinary Least Squares by imposing a penalty on the size of the coefficients. The ridge coefficients minimize a penalized residual sum of squares:

$$\min_{\beta} \|y - X\beta\|^2 + \lambda \|\beta\|^2$$

The complexity parameter $\lambda \geq 0$ controls the amount of shrinkage; the larger the value of λ , the greater the amount of shrinkage, and thus the coefficients become more robust to collinearity.

As with other linear models, Ridge will take in its `fit` method arrays X , y , and will store the coefficients β of the linear model in its `coef` member.

scikitlearn user guide Release 0213

```
from sklearn import linearmodel
```

```
reg = linearmodel.Ridge(alpha=5)
```

```
reg.fit(0 0 0 1 1 0 1 1)
```

```
Ridge(alpha=0.5, copy_X=True, fit_intercept=True, max_iter=None,
```

```
normalize=False, random_state=None, solver='auto', tol=0.001,
```

```
reg.coef
```

```
array(0.34545455 0.34545455
```

```
reg.intercept_
```

```
0.13636
```

Examples

- Plot Ridge coefficients as a function of the regularization

- Classification of text documents using sparse features

Ridge Complexity

This method has the same order of complexity as Ordinary Least Squares

Setting the regularization parameter generalized CrossValidation

RidgeCV implements ridge regression with builtin crossvalidation of the alpha parameter The object works in the same way as GridSearchCV except that it defaults to Generalized CrossValidation GCV an efficient form of leaveoneout crossvalidation

```
import numpy as np
```

```
from sklearn import linearmodel
```

```
reg = linearmodel.RidgeCV(alphas=np.logspace(-6, 6, 13))
```

```
reg.fit(0 0 0 1 1 0 1 1)
```

```
RidgeCV(alphas=[1e-06, 1e-05, 1e-04, 1e-03, 1e-02, 1e-01, 1e00, 1e01,
```

```
1e02, 1e03, 1e04, 1e05, 1e06],
```

```
cv=None, fit_intercept=True, gcv_mode=None, normalize=False,
```

```
scoring=None, store_cv_values=False,
```

```
reg.alpha_
```

```
0.01
```

Specifying the value of the cv attribute will trigger the use of crossvalidation with GridSearchCV for example

cv=10 for 10fold crossvalidation rather than Generalized CrossValidation

References

- “Notes on Regularized Least Squares” Rifkin Lippert technical report course slides

Lasso

The Lasso is a linear model that estimates sparse coefficients It is useful in some contexts due to its tendency to prefer solutions with fewer nonzero coefficients effectively reducing the number of features upon which the given solution is dependent For this reason Lasso and its variants are fundamental to the field of compressed sensing

31 Supervised learning 199

scikitlearn user guide Release 0213

Under certain conditions it can recover the exact set of nonzero coefficients see Compressive sensing tomography reconstruction with L1 prior Lasso

Mathematically it consists of a linear model with an added regularization term The objective function to minimize is

$$\min_{\beta} \frac{1}{2} \sum_{i=1}^n \text{samples}_i - \beta^2 + 2\lambda \|\beta\|_1$$

The lasso estimate thus solves the minimization of the leastsquares penalty with $\lambda \|\beta\|_1$ added where λ is a constant and $\|\beta\|_1$ is the ℓ_1 norm of the coefficient vector

The implementation in the class Lasso uses coordinate descent as the algorithm to fit the coefficients See Least Angle Regression for another implementation

```
from sklearn import linearmodel
reg = linearmodel.Lasso(alpha=0.1)
reg.fit(0 0 1 1 0 1)
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=1000,
normalize=False, positive=False, precompute=False, random_state=None,
selection=cyclic, tol=0.0001, warm_start=False)
reg.predict(1 1)
array(0.8)
```

The function lassopath is useful for lowerlevel tasks as it computes the coefficients along the full path of possible values

Examples

- Lasso and Elastic Net for Sparse Signals
- Compressive sensing tomography reconstruction with L1 prior Lasso

Note Feature selection with Lasso

As the Lasso regression yields sparse models it can thus be used to perform feature selection as detailed in L1based feature selection

The following two references explain the iterations used in the coordinate descent solver of scikitlearn as well as the duality gap computation used for convergence control

References

- “Regularization Path For Generalized linear Models by Coordinate Descent” Friedman Hastie Tibshirani J Stat Softw 2010 Paper
- “An InteriorPoint Method for LargeScale L1Regularized Least Squares” S J Kim K Koh M Lustig S Boyd and D Gorinevsky in IEEE Journal of Selected Topics in Signal Processing 2007 Paper

Setting regularization parameter

The alpha parameter controls the degree of sparsity of the estimated coefficients

200 Chapter 3 User Guide

scikitlearn user guide Release 0213

Using crossvalidation

scikitlearn exposes objects that set the Lasso alpha parameter by crossvalidation LassoCV andLassoLarsCV

LassoLarsCV is based on the Least Angle Regression algorithm explained below

For highdimensional datasets with many collinear features LassoCV is most often preferable However

LassoLarsCV has the advantage of exploring more relevant values of alpha parameter and if the number of samples is very small compared to the number of features it is often faster than LassoCV

Informationcriteria based model selection

Alternatively the estimator LassoLarsIC proposes to use the Akaike information criterion AIC and the Bayes

Information criterion BIC It is a computationally cheaper alternative to find the optimal value of alpha as the regu

larization path is computed only once instead of k1 times when using kfold crossvalidation However such criteria

needs a proper estimation of the degrees of freedom of the solution are derived for large samples asymptotic results

and assume the model is correct ie that the data are actually generated by this model They also tend to break when

the problem is badly conditioned more features than samples

Examples

•Lasso model selection CrossValidation AIC BIC

31 Supervised learning 201

scikitlearn user guide Release 0213

Comparison with the regularization parameter of SVM (Support Vector Machine)

The equivalence between α and the regularization parameter of SVM (Support Vector Machine) C is given by $\alpha = 1/C$ or $\alpha = 1/(n \cdot \text{samples} \cdot C)$ depending on the estimator and the exact objective function optimized by the model

Multitask Lasso

The `MultiTaskLasso` is a linear model that estimates sparse coefficients for multiple regression problems jointly. y is a 2D array of shape $(n_{\text{samples}}, n_{\text{tasks}})$. The constraint is that the selected features are the same for all the regression problems also called tasks

The following figure compares the location of the nonzero entries in the coefficient matrix W obtained with a simple Lasso or a `MultiTaskLasso`. The Lasso estimates yield scattered nonzeros while the nonzeros of the `MultiTaskLasso` are full columns

Fitting a timeseries model imposing that any active feature be active at all times

Examples

- Joint feature selection with multitask Lasso

Mathematically it consists of a linear model trained with a mixed ℓ_1/ℓ_2 norm for regularization. The objective function

202 Chapter 3 User Guide

to minimize is

min

$\frac{1}{2}$

$\sum_{i=1}^n \text{samples}_i - \frac{1}{2}$

Fro^2

where Fro indicates the Frobenius norm

$\text{Fro} \sqrt{\sum}$

\sum^2

\sum

and l_1/l_2 reads

$\frac{1}{2} \sum$

$\sqrt{\sum}$

\sum^2

\sum

The implementation in the class MultiTaskLasso uses coordinate descent as the algorithm to fit the coefficients

ElasticNet

ElasticNet is a linear regression model trained with both l_1 and l_2 norm regularization of the coefficients This combination allows for learning a sparse model where few of the weights are nonzero like Lasso while still maintaining the regularization properties of Ridge We control the convex combination of l_1 and l_2 using the l_1 ratio parameter

Elasticnet is useful when there are multiple features which are correlated with one another Lasso is likely to pick one of these at random while elasticnet is likely to pick both

A practical advantage of trading off between Lasso and Ridge is that it allows ElasticNet to inherit some of Ridge's stability under rotation

The objective function to minimize is in this case

min

$\frac{1}{2}$

$\sum_{i=1}^n \text{samples}_i - \frac{1}{2}$

$\sum \alpha \sum^2$

\sum^2

\sum

The class ElasticNetCV can be used to set the parameters α and l_1 ratio α by crossvalidation

scikitlearn user guide Release 0213

Examples

- Lasso and Elastic Net for Sparse Signals
- Lasso and Elastic Net

The following two references explain the iterations used in the coordinate descent solver of scikitlearn as well as the duality gap computation used for convergence control

References

- “Regularization Path For Generalized linear Models by Coordinate Descent” Friedman Hastie Tibshirani J Stat Softw 2010 Paper
- “An InteriorPoint Method for LargeScale L1Regularized Least Squares” S J Kim K Koh M Lustig S Boyd and D Gorinevsky in IEEE Journal of Selected Topics in Signal Processing 2007 Paper

Multitask ElasticNet

TheMultiTaskElasticNet is an elasticnet model that estimates sparse coefficients for multiple regression problems jointly Y is a 2D array of shape nsamples ntasks The constraint is that the selected features are the same for all the regression problems also called tasks

Mathematically it consists of a linear model trained with a mixed ℓ_1/ℓ_2 norm and ℓ_2 norm for regularization The objective function to minimize is

$$\min_{\beta} \frac{1}{2} \sum_{i=1}^n \text{samples}_i - \frac{1}{2} \sum_{j=1}^n \text{tasks}_j^2 + \lambda \left(\sum_{j=1}^n \|\beta_j\|_1 + \frac{1}{2} \sum_{j=1}^n \|\beta_j\|_2^2 \right)$$

The implementation in the class MultiTaskElasticNet uses coordinate descent as the algorithm to fit the coefficients

The classMultiTaskElasticNetCV can be used to set the parameters alpha andl1ratio by cross validation

Least Angle Regression

Leastangle regression LARS is a regression algorithm for highdimensional data developed by Bradley Efron Trevor Hastie Iain Johnstone and Robert Tibshirani LARS is similar to forward stepwise regression At each step it finds the feature most correlated with the target When there are multiple features having equal correlation instead of continuing along the same feature it proceeds in a direction equiangular between the features

The advantages of LARS are

- It is numerically efficient in contexts where the number of features is significantly greater than the number of samples
- It is computationally just as fast as forward selection and has the same order of complexity as ordinary least squares
- It produces a full piecewise linear solution path which is useful in crossvalidation or similar attempts to tune the model
- If two features are almost equally correlated with the target then their coefficients should increase at approximately the same rate The algorithm thus behaves as intuition would expect and also is more stable

scikitlearn user guide Release 0213

- It is easily modified to produce solutions for other estimators like the Lasso

The disadvantages of the LARS method include

- Because LARS is based upon an iterative refitting of the residuals it would appear to be especially sensitive to the effects of noise This problem is discussed in detail by Weisberg in the discussion section of the Efron et al 2004 Annals of Statistics article

The LARS model can be used using estimator `Lars` or its lowlevel implementation `larspath` or

`larspathgram`

`LARS Lasso`

`LassoLars` is a lasso model implemented using the LARS algorithm and unlike the implementation based on coordinate descent this yields the exact solution which is piecewise linear as a function of the norm of its coefficients

```
from sklearn import linearmodel
```

```
reg = linearmodel.LassoLars(alpha=1
```

```
regfit0 0 1 1 0 1
```

```
LassoLars(alpha=0.1, copy_X=True, eps=fit_intercept=True
```

```
fit_path=True, max_iter=500, normalize=True, positive=False
```

```
precompute=None, verbose=False
```

```
reg.coef
```

```
array(0.717157, 0)
```

Examples

- Lasso path using LARS

The `Lars` algorithm provides the full path of the coefficients along the regularization parameter almost for free thus a common operation is to retrieve the path with one of the functions `larspath` or `larspathgram`

Mathematical formulation

The algorithm is similar to forward stepwise regression but instead of including features at each step the estimated coefficients are increased in a direction equiangular to each one's correlations with the residual

31 Supervised learning 205

Instead of giving a vector result the LARS solution consists of a curve denoting the solution for each value of the ℓ_1 norm of the parameter vector. The full coefficients path is stored in the array `coefpath` which has size `nfeatures`. The first column is always zero.

References

- Original Algorithm is detailed in the paper Least Angle Regression by Hastie et al

Orthogonal Matching Pursuit OMP

`OrthogonalMatchingPursuit` and `orthogonalmp` implements the OMP algorithm for approximating the fit of a linear model with constraints imposed on the number of nonzero coefficients ie the ℓ_0 pseudonorm

Being a forward feature selection method like Least Angle Regression, orthogonal matching pursuit can approximate the optimum solution vector with a fixed number of nonzero elements

$$\arg \min$$

$$\|y - X\beta\|_2^2$$

$$\text{subject to } 0 \leq \|\beta\|_0 \leq \text{nonzero_coefs}$$

Alternatively orthogonal matching pursuit can target a specific error instead of a specific number of nonzero coefficients. This can be expressed as

$$\arg \min$$

$$\|y - X\beta\|_2^2$$

$$\text{subject to } \|\beta\|_0 \leq \text{tol}$$

OMP is based on a greedy algorithm that includes at each step the atom most highly correlated with the current residual. It is similar to the simpler matching pursuit (MP) method but better in that at each iteration the residual is recomputed using an orthogonal projection on the space of the previously chosen dictionary elements.

Examples

- `Orthogonal Matching Pursuit`

References

- <https://www.cse.cmu.edu/~acilron/rubinpapers/KSVDOMPv2.pdf>
- Matching pursuits with time-frequency dictionaries S. G. Mallat, Z. Zhang

Bayesian Regression

Bayesian regression techniques can be used to include regularization parameters in the estimation procedure. The regularization parameter is not set in a hard sense but tuned to the data at hand.

This can be done by introducing uninformative priors over the hyper parameters of the model. The ℓ_2 regularization used in Ridge Regression is equivalent to finding a maximum a posteriori estimation under a Gaussian prior over the coefficients β with precision λ^{-1} . Instead of setting λ manually it is possible to treat it as a random variable to be estimated from the data.

To obtain a fully probabilistic model the output y is assumed to be Gaussian distributed around $X\beta$.

$$y \sim \mathcal{N}(X\beta, \sigma^2 I)$$

scikitlearn user guide Release 0.21.3

where β is again treated as a random variable that is to be estimated from the data

The advantages of Bayesian Regression are

- It adapts to the data at hand
- It can be used to include regularization parameters in the estimation procedure

The disadvantages of Bayesian regression include

- Inference of the model can be time consuming

References

- A good introduction to Bayesian methods is given in C Bishop Pattern Recognition and Machine learning
- Original Algorithm is detailed in the book Bayesian learning for neural networks by Radford M Neal

Bayesian Ridge Regression

BayesianRidge estimates a probabilistic model of the regression problem as described above. The prior for the coefficient β is given by a spherical Gaussian

$$p(\beta) \propto \exp\left(-\frac{1}{2}\beta^T\Lambda\beta\right)$$

The priors over α and σ^2 are chosen to be gamma distributions the conjugate prior for the precision of the Gaussian

The resulting model is called Bayesian Ridge Regression and is similar to the classical Ridge

The parameters α and σ^2 are estimated jointly during the fit of the model the regularization parameters α and σ^2 being estimated by maximizing the log marginal likelihood. The scikitlearn implementation is based on the algorithm described in Appendix A of Tipping 2001 where the update of the parameters α and σ^2 is done as suggested in MacKay 1992.

The remaining hyperparameters are the parameters α_1 and α_2 of the gamma priors over α and σ^2 . These are usually chosen to be noninformative. By default $\alpha_1 = 10^{-6}$ and $\alpha_2 = 10^{-6}$.

Bayesian Ridge Regression is used for regression

scikitlearn user guide Release 0213

```
from sklearn import linearmodel
```

```
X = [[0, 0, 1, 1, 2, 2, 3, 3]
```

```
Y = [0, 1, 2, 3]
```

```
reg = linearmodel.BayesianRidge
```

```
reg.fit(X, Y)
```

```
BayesianRidge(alpha=1e-06, alpha2=1e-06, compute_score=False, copy_X=True,
```

```
fit_intercept=True, lambda_1=1e-06, lambda_2=1e-06, n_iter=300,
```

```
normalize=False, tol=0.001, verbose=False)
```

After being fitted the model can then be used to predict new values

```
reg.predict([1, 0])
```

```
array([0.5, 0.0000013])
```

The coefficients of the model can be accessed

```
reg.coef_
```

```
array([0.499999993, 0.499999993])
```

Due to the Bayesian framework the weights found are slightly different to the ones found by Ordinary Least Squares

However Bayesian Ridge Regression is more robust to illposed problems

Examples

- Bayesian Ridge Regression

References

- Section 33 in Christopher M Bishop Pattern Recognition and Machine Learning 2006

- David J C MacKay Bayesian Interpolation 1992

- Michael E Tipping Sparse Bayesian Learning and the Relevance Vector Machine 2001

Automatic Relevance Determination (ARD)

ARDRegression is very similar to Bayesian Ridge Regression but can lead to sparser coefficients

ARDRegression poses a different prior over β by dropping the assumption of the Gaussian being spherical

Instead the distribution over β is assumed to be an axisparallel elliptical Gaussian distribution

This means each coefficient β_j is drawn from a Gaussian distribution centered on zero and with a precision λ_j

$\lambda_j \sim \text{Gamma}(\alpha, \beta)$

with $\text{diag}(\lambda) = \text{diag}(\lambda_1, \dots, \lambda_p)$

In contrast to Bayesian Ridge Regression each coordinate of β has its own standard deviation λ_j . The prior over all

β is chosen to be the same gamma distribution given by hyperparameters α and β

ARD is also known in the literature as Sparse Bayesian Learning and Relevance Vector Machine

1 Christopher M Bishop Pattern Recognition and Machine Learning Chapter 7.2.1

2 David Wipf and Srikantan Nagarajan A new view of automatic relevance determination

3 Michael E Tipping Sparse Bayesian Learning and the Relevance Vector Machine

208 Chapter 3 User Guide

Examples

- Automatic Relevance Determination Regression ARD

References

Logistic regression

Logistic regression despite its name is a linear model for classification rather than regression Logistic regression is also known in the literature as logit regression maximumentropy classification MaxEnt or the loglinear classifier In this model the probabilities describing the possible outcomes of a single trial are modeled using a logistic function Logistic regression is implemented in LogisticRegression This implementation can fit binary OnevsRest or multinomial logistic regression with optional ℓ_1 / ℓ_2 or ElasticNet regularization Note Regularization is applied by default which is common in machine learning but not in statistics Another advantage of regularization is that it improves numerical stability No regularization amounts to setting C to a very high value

As an optimization problem binary class ℓ_2 penalized logistic regression minimizes the following cost function

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|_2^2 - \sum_{i=1}^n \log \exp(\mathbf{w}^T \mathbf{x}_i) + \sum_{i=1}^n \log \exp(\mathbf{w}^T \mathbf{x}_i)$$

Similarly ℓ_1 regularized logistic regression solves the following optimization problem

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|_1 - \sum_{i=1}^n \log \exp(\mathbf{w}^T \mathbf{x}_i) + \sum_{i=1}^n \log \exp(\mathbf{w}^T \mathbf{x}_i)$$

ElasticNet regularization is a combination of ℓ_1 and ℓ_2 and minimizes the following cost function

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_1 - \sum_{i=1}^n \log \exp(\mathbf{w}^T \mathbf{x}_i) + \sum_{i=1}^n \log \exp(\mathbf{w}^T \mathbf{x}_i)$$

where α controls the strength of l_1 regularization vs l_2 regularization it corresponds to the `l1_ratio` parameter

Note that in this notation it's assumed that the target y takes values in the set $\{-1, 1\}$ at trial i . We can also see that ElasticNet is equivalent to l_1 when $\alpha = 1$ and equivalent to l_2 when $\alpha = 0$

The solvers implemented in the class `LogisticRegression` are "liblinear" "newtoncg" "lbfgs" "sag" and "saga"

The solver "liblinear" uses a coordinate descent CD algorithm and relies on the excellent C LIBLINEAR library which is shipped with scikitlearn. However the CD algorithm implemented in liblinear cannot learn a true multinomial multiclass model instead the optimization problem is decomposed in a "onevsrest" fashion so separate binary classifiers are trained for all classes. This happens under the hood so `LogisticRegression` instances using this solver behave as multiclass classifiers. For l_1 regularization `sklearnsvm.l1minc` allows to calculate the lower bound for C in order to get a non "null" all feature weights to zero model.

The "lbfgs" "sag" and "newtoncg" solvers only support l_2 regularization or no regularization and are found to converge faster for some highdimensional data. Setting multiclass to "multinomial" with these solvers learns a true multinomial logistic regression model⁵ which means that its probability estimates should be better calibrated than the default "onevsrest" setting.

The "sag" solver uses Stochastic Average Gradient descent⁶. It is faster than other solvers for large datasets when both the number of samples and the number of features are large.

The "saga" solver⁷ is a variant of "sag" that also supports the nonsmooth penalty l_1 . This is therefore the solver of choice for sparse multinomial logistic regression. It is also the only solver that supports `penalty=elasticnet`.

The "lbfgs" is an optimization algorithm that approximates the Broyden-Fletcher-Goldfarb-Shanno algorithm⁸ which belongs to quasi-Newton methods. The "lbfgs" solver is recommended for use for small datasets but for larger datasets its performance suffers⁹.

The following table summarizes the penalties supported by each solver.

| Solvers | Penalties | 'liblinear' | 'lbfgs' | 'newtoncg' | 'sag' | 'saga' |
|-------------|---------------|-------------|---------|------------|-------|--------|
| Multinomial | L_2 penalty | no | yes | yes | yes | yes |
| OVR | L_2 penalty | yes | yes | yes | yes | yes |
| Multinomial | L_1 penalty | no | no | no | no | yes |
| OVR | L_1 penalty | yes | no | no | no | yes |
| ElasticNet | no | no | no | no | yes | |
| No penalty | 'none' | no | yes | yes | yes | yes |

Behaviors

| | | | | | | |
|-----------------------------|-----|-----|-----|-----|-----|----|
| Penalize the intercept | bad | yes | no | no | no | no |
| Faster for large datasets | no | no | no | yes | yes | |
| Robust to unscaled datasets | yes | yes | yes | no | no | |

The "lbfgs" solver is used by default for its robustness. For large datasets the "saga" solver is usually faster. For large dataset you may also consider using `SGDClassifier` with 'log' loss which might be even faster but requires more tuning.

⁵Christopher M Bishop Pattern Recognition and Machine Learning Chapter 4.3.4
⁶Mark Schmidt, Nicolas Le Roux and Francis Bach Minimizing Finite Sums with the Stochastic Average Gradient
⁷Aaron Defazio, Francis Bach, Simon Lacoste-Julien SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives
⁸https://en.wikipedia.org/wiki/Broyden%E2%80%93Fletcher%E2%80%93Goldfarb%E2%80%93Shanno_algorithm
⁹"Performance Evaluation of Lbfgs vs other solvers"
2.10 Chapter 3 User Guide

scikitlearn user guide Release 0213

Examples

- L1 Penalty and Sparsity in Logistic Regression
- Regularization path of L1 Logistic Regression
- Plot multinomial and OnevsRest Logistic Regression
- Multiclass sparse logisitic regression on newgroups20
- MNIST classification using multinomial logistic L1

Differences from liblinear

There might be a difference in the scores obtained between LogisticRegression withsolverliblinear orLinearSVC and the external liblinear library directly when fitinterceptFalse and the fit coef or the data to be predicted are zeroes This is because for the samples with decisionfunction zero LogisticRegression andLinearSVC predict the negative class while liblinear predicts the positive class Note that a model with fitinterceptFalse and having many samples with decisionfunction zero is likely to be a underfit bad model and you are advised to set fitinterceptTrue and increase the inter ceptscaling

Note Feature selection with sparse logistic regression

A logistic regression with l1penalty yields sparse models and can thus be used to perform feature selection as detailed in L1based feature selection

LogisticRegressionCV implements Logistic Regression with builtin crossvalidation support to find the opti malCandl1ratio parameters according to the scoring attribute The “newtoncg” “sag” “saga” and “lbfgs” solvers are found to be faster for highdimensional dense data due to warmstarting see Glossary

References

Stochastic Gradient Descent SGD

Stochastic gradient descent is a simple yet very efficient approach to fit linear models It is particularly useful when the number of samples and the number of features is very large The partialfit method allows onlineoutofcore learning

The classes SGDClassifier andSGDRegressor provide functionality to fit linear models for classifica tion and regression using different convex loss functions and different penalties Eg with losslog SGDClassifier fits a logistic regression model while with lossHINGE it fits a linear support vector ma chine SVM (Support Vector Machine)

References

- Stochastic Gradient Descent

scikitlearn user guide Release 0.21.3

Perceptron

The Perceptron is another simple classification algorithm suitable for large scale learning. By default

- It does not require a learning rate
- It is not regularized/penalized
- It updates its model only on mistakes

The last characteristic implies that the Perceptron is slightly faster to train than SGD with the hinge loss and that the resulting models are sparser.

Passive Aggressive Algorithms

The passive aggressive algorithms are a family of algorithms for large scale learning. They are similar to the Perceptron in that they do not require a learning rate. However, contrary to the Perceptron, they include a regularization parameter C .

For classification, `PassiveAggressiveClassifier` can be used with `loss=hinge` (PAI) or `loss=squared_hinge` (PAII). For regression, `PassiveAggressiveRegressor` can be used with `loss=epsilon_insensitive` (PAI) or `loss=squared_epsilon_insensitive` (PAII).

References

- “Online Passive Aggressive Algorithms” K. Crammer, O. Dekel, J. Keshat, S. Shalev-Shwartz, Y. Singer
JMLR 7 2006

Robustness: regression, outliers and modeling errors

Robust regression aims to fit a regression model in the presence of corrupt data, either outliers or error in the model. Different scenarios and useful concepts.

There are different things to keep in mind when dealing with data corrupted by outliers.

scikitlearn user guide Release 0213

- Outliers in X or in y
- Outliers in the y direction Outliers in the X direction
- Fraction of outliers versus amplitude of error

The number of outlying points matters but also how much they are outliers

Small outliers Large outliers

An important notion of robust fitting is that of breakdown point the fraction of data that can be outlying for the fit to start missing the inlying data

Note that in general robust fitting in highdimensional setting large nfeatures is very hard The robust models here will probably not work in these settings

Tradeoffs which estimator

Scikitlearn provides 3 robust regression estimators RANSAC Theil Sen andHuberRegressor

- HuberRegressor should be faster than RANSAC andTheil Sen unless the number of samples are very large ie nsamples nfeatures This is because RANSAC andTheil Sen fit on smaller subsets of the data However both Theil Sen andRANSAC are unlikely to be as robust as HuberRegressor for the default parameters

31 Supervised learning 213

scikitlearn user guide Release 0213

- RANSAC is faster than Theil Sen and scales much better with the number of samples
- RANSAC will deal better with large outliers in the y direction most common situation
- Theil Sen will cope better with mediusize outliers in the X direction but this property will disappear in highdimensional settings

When in doubt use RANSAC

RANSAC RANdom SAmple Consensus

RANSAC RANdom SAmple Consensus fits a model from random subsets of inliers from the complete data set

RANSAC is a nondeterministic algorithm producing only a reasonable result with a certain probability which is dependent on the number of iterations see maxtrials parameter It is typically used for linear and nonlinear regression problems and is especially popular in the field of photogrammetric computer vision

The algorithm splits the complete input sample data into a set of inliers which may be subject to noise and outliers which are eg caused by erroneous measurements or invalid hypotheses about the data The resulting model is then estimated only from the determined inliers

Details of the algorithm

Each iteration performs the following steps

1 Selectminsamples random samples from the original data and check whether the set of data is valid see isdatavalid

2 Fit a model to the random subset baseestimatorfit and check whether the estimated model is valid seeismodelvalid

3 Classify all data as inliers or outliers by calculating the residuals to the estimated model baseestimator predictX y all data samples with absolute residuals smaller than the residualthreshold are considered as inliers

4 Save fitted model as best model if number of inlier samples is maximal In case the current estimated model has the same number of inliers it is only considered as the best model if it has better score

scikitlearn user guide Release 0213

These steps are performed either a maximum number of times `maxtrials` or until one of the special stop criteria are met see `stopninliers` and `stopscore`. The final model is estimated using all inlier samples consensus set of the previously determined best model

The `isdatavalid` and `ismodelvalid` functions allow to identify and reject degenerate combinations of random subsamples. If the estimated model is not needed for identifying degenerate cases `isdatavalid` should be used as it is called prior to fitting the model and thus leading to better computational performance

- Examples
- Robust linear model estimation using RANSAC
 - Robust linear estimator fitting

References

- <https://en.wikipedia.org/wiki/RANSAC>
- “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography” Martin A Fischler and Robert C Bolles. SRI International 1981
- “Performance Evaluation of RANSAC Family” Sunglok Choi Taemin Kim and Wonpil Yu. BMVC 2009

TheilSen estimator generalized median based estimator

The TheilSenRegressor estimator uses a generalization of the median in multiple dimensions. It is thus robust to multivariate outliers. Note however that the robustness of the estimator decreases quickly with the dimensionality of the problem. It loses its robustness properties and becomes no better than an ordinary least squares in high dimension

- Examples
- TheilSen Regression
 - Robust linear estimator fitting

References

- https://en.wikipedia.org/wiki/Theil%E2%80%93S%C3%A9n_estimator

Theoretical considerations

TheilSenRegressor is comparable to the Ordinary Least Squares (OLS) in terms of asymptotic efficiency and as an unbiased estimator. In contrast to OLS, TheilSen is a nonparametric method which means it makes no assumption about the underlying distribution of the data. Since TheilSen is a median based estimator, it is more robust against corrupted data aka outliers. In univariate setting, TheilSen has a breakdown point of about 29.3% in case of a simple linear regression which means that it can tolerate arbitrary corrupted data of up to 29.3%.

31 Supervised learning 215

scikitlearn user guide Release 0213

The implementation of TheilSenRegressor in scikitlearn follows a generalization to a multivariate linear regression model using the spatial median which is a generalization of the median to multiple dimensions. In terms of time and space complexity TheilSen scales according to

\sqrt{n} samples
 \sqrt{n} subsamples
which makes it infeasible to be applied exhaustively to problems with a large number of samples and features. Therefore the magnitude of a subpopulation can be chosen to limit the time and space complexity by considering only a random subset of all possible combinations.

Examples
•TheilSen Regression

References

Huber Regression

TheHuberRegressor is different to Ridge because it applies a linear loss to samples that are classified as outliers. A sample is classified as an inlier if the absolute error of that sample is lesser than a certain threshold. It differs from TheilSenRegressor andRANSACRegressor because it does not ignore the effect of the outliers but gives a lesser weight to them.

The loss function that HuberRegressor minimizes is given by

$$\min_{\beta} \sum_{i=1}^n \rho(r_i)$$
$$\rho(r) = \begin{cases} \frac{1}{2}r^2 & \text{if } |r| \leq c \\ c|r| - \frac{1}{2}c^2 & \text{if } |r| > c \end{cases}$$

where $c = \sqrt{2}$

10Xin Dang Hanxiang Peng Xueqin Wang and Heping Zhang TheilSen Estimators in a Multiple Linear Regression Model
11
20 Kärkkäinen and S Äyrämö On Computation of Spatial Median for Robust Data Mining
216 Chapter 3 User Guide

scikitlearn user guide Release 0.21.3

where

ϵ

ϵ^2 if $\epsilon > 2$

$2\epsilon - \epsilon^2$ otherwise

It is advised to set the parameter epsilon to 1.35 to achieve 95 statistical efficiency

Notes

The HuberRegressor differs from using SGDRegressor with loss set to huber in the following ways

- HuberRegressor is scaling invariant. Once epsilon is set, scaling X and y down or up by different values would produce the same robustness to outliers as before, as compared to SGDRegressor where epsilon has to be set again when X and y are scaled.
- HuberRegressor should be more efficient to use on data with small number of samples while SGDRegressor needs a number of passes on the training data to produce the same robustness.

Examples

- HuberRegressor vs Ridge on dataset with strong outliers

References

- Peter J. Huber, Elvezio M. Ronchetti, Robust Statistics, Concomitant scale estimates, pg. 172

Note that this estimator is different from the R implementation of Robust Regression <http://www.stat.columbia.edu/robust/> because the R implementation does a weighted least squares implementation with weights given to each sample on the basis of how much the residual is greater than a certain threshold.

Polynomial regression extending linear models with basis functions

One common pattern within machine learning is to use linear models trained on nonlinear functions of the data. This approach maintains the generally fast performance of linear methods while allowing them to fit a much wider range of data.

31 Supervised learning 217

scikitlearn user guide Release 0213

For example a simple linear regression can be extended by constructing polynomial features from the coefficients
In the standard linear regression case you might have a model that looks like this for twodimensional data

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

If we want to fit a paraboloid to the data instead of a plane we can combine the features in secondorder polynomials
so that the model looks like this

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2$$

The sometimes surprising observation is that this is still a linear model to see this imagine creating a new set of
features

$$\begin{bmatrix} 1 & x_1 & x_2 & x_1^2 & x_2^2 & x_1 x_2 \end{bmatrix}$$

With this relabeling of the data our problem can be written

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2$$

We see that the resulting polynomial regression is in the same class of linear models we considered above ie the
model is linear in θ and can be solved by the same techniques By considering linear fits within a higherdimensional
space built with these basis functions the model has the flexibility to fit a much broader range of data

Here is an example of applying this idea to onedimensional data using polynomial features of varying degrees

This figure is created using the PolynomialFeatures transformer which transforms an input data matrix into a
new data matrix of a given degree It can be used as follows

```
from sklearn.preprocessing import PolynomialFeatures
import numpy as np
X = np.arange(6).reshape(3, 2)
```

```
X
array([[0, 1],
       [2, 3],
       [4, 5]])
poly = PolynomialFeatures(degree=2)
poly.fit_transform(X)
array([[1, 0, 1, 0, 0, 1],
       [1, 2, 3, 4, 6, 9],
       [1, 4, 5, 16, 20, 25]])
```


scikitlearn user guide Release 0213

The features of X have been transformed from $\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$ to $\begin{bmatrix} 1 & 2 & 1 \\ 1 & 2 & 4 \end{bmatrix}$

$\begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$

2 and can now be used within any

linear model

This sort of preprocessing can be streamlined with the Pipeline tools A single object representing a simple polynomial

regression can be created and used as follows

```
from sklearn.preprocessing import PolynomialFeatures
```

```
from sklearn.linear_model import LinearRegression
```

```
from sklearn.pipeline import Pipeline
```

```
import numpy as np
```

```
model = Pipeline([PolynomialFeatures(degree=3,
```

```
linear=LinearRegression(fit_intercept=False,
```

```
fit_to_an_order_3_polynomial_data
```

```
x=np.arange(5,
```

```
y=[3, 2, x**2, x**3,
```

```
model=model.fit(x, np.newaxis * y)
```

```
model.named_steps['linear'].coef_
```

```
array([ 3.  2.  1.  1.]
```

The linear model trained on polynomial features is able to exactly recover the input polynomial coefficients

In some cases it's not necessary to include higher powers of any single feature but only the so called interaction

features that multiply together at most ≤ 2 distinct features These can be gotten from PolynomialFeatures with

the setting `interaction_only=True`

For example when dealing with boolean features $\begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$

$\begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$ for all $\begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$ and is therefore useless but $\begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$ represents the

conjunction of two booleans This way we can solve the XOR problem with a linear classifier

```
from sklearn.linear_model import Perceptron
```

```
from sklearn.preprocessing import PolynomialFeatures
```

```
import numpy as np
```

```
X=np.array([0, 0, 1, 1, 0, 1, 1])
```

```
y=[X[0], X[1],
```

```
0, 1, 0, 1, 0]
```

```
X=PolynomialFeatures(interaction_only=True).fit_transform(X).astype(int)
```

```
X
```

```
array([ 0,  0,  0,  0,
```

```
1,  0,  1,  0,
```

```
1,  1,  0,  0,
```

```
1,  1,  1,  1])
```

```
clf=Perceptron(fit_intercept=False, max_iter=10, tol=None,
```

```
shuffle=False).fit(X, y)
```

And the classifier "predictions" are perfect

```
clf.predict(X)
```

```
array([ 0,  1,  1,  0,
```

```
0,  1,  1,  0])
```

```
clf.score(X, y)
```

```
1.0
```

312 Linear and Quadratic Discriminant Analysis

Linear Discriminant Analysis `discriminantanalysis.LinearDiscriminantAnalysis` and

Quadratic Discriminant Analysis `discriminantanalysis.QuadraticDiscriminantAnalysis`

are two classic classifiers with as their names suggest a linear and a quadratic decision surface respectively

31 Supervised learning 219

scikitlearn user guide Release 0213

These classifiers are attractive because they have closedform solutions that can be easily computed are inherently multiclass have proven to work well in practice and have no hyperparameters to tune

The plot shows decision boundaries for Linear Discriminant Analysis and Quadratic Discriminant Analysis The bottom row demonstrates that Linear Discriminant Analysis can only learn linear boundaries while Quadratic Discriminant Analysis can learn quadratic boundaries and is therefore more flexible

Examples

Linear and Quadratic Discriminant Analysis with covariance ellipsoid Comparison of LDA and QDA on synthetic data

Dimensionality reduction using Linear Discriminant Analysis

discriminantanalysisLinearDiscriminantAnalysis can be used to perform supervised dimensionality reduction by projecting the input data to a linear subspace consisting of the directions which maximize the separation between classes in a precise sense discussed in the mathematics section below The dimension of the output is necessarily less than the number of classes so this is in general a rather strong dimensionality reduction and only makes sense in a multiclass setting

220 Chapter 3 User Guide

scikitlearn user guide Release 0213

This is implemented in `discriminantanalysisLinearDiscriminantAnalysis` transform. The desired dimensionality can be set using the `ncomponents` constructor parameter. This parameter has no influence on `discriminantanalysisLinearDiscriminantAnalysis` fit or `discriminantanalysisLinearDiscriminantAnalysis` predict.

Examples

Comparison of LDA and PCA (Principal Component Analysis) 2D projection of Iris dataset. Comparison of LDA and PCA (Principal Component Analysis) reduction of the Iris dataset.

Mathematical formulation of the LDA and QDA classifiers

Both LDA and QDA can be derived from simple probabilistic models which model the class conditional distribution of the data \mathbf{x} for each class ω . Predictions can then be obtained by using Bayes' rule

$$p(\omega|\mathbf{x}) = \frac{p(\mathbf{x}|\omega)p(\omega)}{p(\mathbf{x})}$$

$$p(\mathbf{x}|\omega) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right\}$$

$$p(\omega) = \frac{1}{N_\omega}$$

and we select the class ω which maximizes this conditional probability

More specifically for linear and quadratic discriminant analysis \mathbf{x} is modeled as a multivariate Gaussian distribution with density

$$p(\mathbf{x}|\omega) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right\}$$

$$p(\omega) = \frac{1}{N_\omega}$$

$$p(\omega|\mathbf{x}) = \frac{p(\mathbf{x}|\omega)p(\omega)}{p(\mathbf{x})}$$

$$p(\omega|\mathbf{x}) = \frac{p(\mathbf{x}|\omega)p(\omega)}{p(\mathbf{x})}$$

where D is the number of features

To use this model as a classifier we just need to estimate from the training data the class priors $p(\omega)$ by the proportion of instances of class ω , the class means μ by the empirical sample class means and the covariance matrices either by the empirical sample class covariance matrices or by a regularized estimator see the section on shrinkage below.

In the case of LDA the Gaussians for each class are assumed to share the same covariance matrix Σ for all ω . This leads to linear decision surfaces which can be seen by comparing the logprobability ratios $\log\frac{p(\omega|\mathbf{x})}{p(\omega'|\mathbf{x})}$

$$\log\frac{p(\omega|\mathbf{x})}{p(\omega'|\mathbf{x})} = \log\frac{p(\mathbf{x}|\omega)p(\omega)}{p(\mathbf{x}|\omega')p(\omega')}$$

$$\log\frac{p(\omega|\mathbf{x})}{p(\omega'|\mathbf{x})} = \log\frac{p(\mathbf{x}|\omega)p(\omega)}{p(\mathbf{x}|\omega')p(\omega')}$$

$$\log\frac{p(\omega|\mathbf{x})}{p(\omega'|\mathbf{x})} = \log\frac{p(\mathbf{x}|\omega)p(\omega)}{p(\mathbf{x}|\omega')p(\omega')}$$

$$\log\frac{p(\omega|\mathbf{x})}{p(\omega'|\mathbf{x})} = \log\frac{p(\mathbf{x}|\omega)p(\omega)}{p(\mathbf{x}|\omega')p(\omega')}$$

$$\log\frac{p(\omega|\mathbf{x})}{p(\omega'|\mathbf{x})} = \log\frac{p(\mathbf{x}|\omega)p(\omega)}{p(\mathbf{x}|\omega')p(\omega')}$$

$$0 \Leftrightarrow$$

$$\frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right\} \frac{1}{N_\omega} = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu')^\top \Sigma^{-1}(\mathbf{x}-\mu')\right\} \frac{1}{N_{\omega'}}$$

$$\frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right\} \frac{1}{N_\omega} = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu')^\top \Sigma^{-1}(\mathbf{x}-\mu')\right\} \frac{1}{N_{\omega'}}$$

$$\frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right\} \frac{1}{N_\omega} = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu')^\top \Sigma^{-1}(\mathbf{x}-\mu')\right\} \frac{1}{N_{\omega'}}$$

$$\frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right\} \frac{1}{N_\omega} = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu')^\top \Sigma^{-1}(\mathbf{x}-\mu')\right\} \frac{1}{N_{\omega'}}$$

$$\frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right\} \frac{1}{N_\omega} = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu')^\top \Sigma^{-1}(\mathbf{x}-\mu')\right\} \frac{1}{N_{\omega'}}$$

In the case of QDA there are no assumptions on the covariance matrices Σ of the Gaussians leading to quadratic decision surfaces. See 3 for more details.

Note Relation with Gaussian Naive Bayes

If in the QDA model one assumes that the covariance matrices are diagonal then the inputs are assumed to be conditionally independent in each class and the resulting classifier is equivalent to the Gaussian Naive Bayes classifier `naivebayesGaussianNB`.

3 "The Elements of Statistical Learning" Hastie T Tibshirani R Friedman J Section 43 p106119 2008

31 Supervised learning 221

Mathematical formulation of LDA dimensionality reduction

To understand the use of LDA in dimensionality reduction it is useful to start with a geometric reformulation of the LDA classification rule explained above We write K for the total number of target classes Since in LDA we assume that all classes have the same estimated covariance Σ we can rescale the data so that this covariance is the identity $\Sigma = I$ with $\tilde{x} = \Sigma^{-1/2}x$

Then one can show that to classify a data point after scaling is equivalent to finding the estimated class mean $\tilde{\mu}_k$ which is closest to the data point in the Euclidean distance But this can be done just as well after projecting on the $(K-1)$ -dimensional affine subspace Π generated by all the $\tilde{\mu}_k$ for all classes This shows that implicit in the LDA classifier there is a dimensionality reduction by linear projection onto a $(K-1)$ -dimensional space

We can reduce the dimension even more to a chosen d by projecting onto the linear subspace Π_d which maximizes the variance of the d components after projection in effect we are doing a form of PCA (Principal Component Analysis) for the transformed class means $\tilde{\mu}_k$

This d corresponds to the `ncomponents` parameter used in the `LinearDiscriminantAnalysis` transform method See [3](#) for more details

Shrinkage
Shrinkage is a tool to improve estimation of covariance matrices in situations where the number of training samples is small compared to the number of features In this scenario the empirical sample covariance is a poor estimator Shrinkage LDA can be used by setting the shrinkage parameter of the `LinearDiscriminantAnalysis` class to 'auto' This automatically determines the optimal shrinkage parameter in an analytic way following the lemma introduced by Ledoit and Wolf⁴ Note that currently shrinkage only works when setting the solver parameter to 'lsqr' or 'eigen'

The shrinkage parameter can also be manually set between 0 and 1 In particular a value of 0 corresponds to no shrinkage which means the empirical covariance matrix will be used and a value of 1 corresponds to complete shrinkage which means that the diagonal matrix of variances will be used as an estimate for the covariance matrix Setting this parameter to a value between these two extrema will estimate a shrunk version of the covariance matrix

⁴Ledoit O Wolf M Honey I Shrank the Sample Covariance Matrix The Journal of Portfolio Management 30(4) 110-119 2004

Estimation algorithms

The default solver is ‘svd’ It can perform both classification and transform and it does not rely on the calculation of the covariance matrix This can be an advantage in situations where the number of features is large However the ‘svd’ solver cannot be used with shrinkage

The ‘lsqr’ solver is an efficient algorithm that only works for classification It supports shrinkage

The ‘eigen’ solver is based on the optimization of the between class scatter to within class scatter ratio It can be used for both classification and transform and it supports shrinkage However the ‘eigen’ solver needs to compute the covariance matrix so it might not be suitable for situations with a high number of features

Examples

Normal and Shrinkage Linear Discriminant Analysis for classification Comparison of LDA classifiers with and without shrinkage

References

313 Kernel ridge regression

Kernel ridge regression KRR M2012 combines Ridge Regression linear least squares with l2norm regularization with the kernel trick It thus learns a linear function in the space induced by the respective kernel and the data For nonlinear kernels this corresponds to a nonlinear function in the original space

The form of the model learned by KernelRidge is identical to support vector regression SVR (Support Vector Regression) However different loss functions are used KRR uses squared error loss while support vector regression uses ϵ -insensitive loss both combined with l2 regularization In contrast to SVR (Support Vector Regression) fitting KernelRidge can be done in closedform and is typically faster for medium sized datasets On the other hand the learned model is nonsparse and thus slower than SVR (Support Vector Regression) which learns a sparse model for $\epsilon = 0$ at prediction time

The following figure compares KernelRidge and SVR on an artificial dataset which consists of a sinusoidal target function and strong noise added to every fifth datapoint The learned model of KernelRidge and SVR is plotted where both complexity regularization and bandwidth of the RBF (Radial Basis Function) kernel have been optimized using gridsearch The learned functions are very similar however fitting KernelRidge is approx seven times faster than fitting SVR (Support Vector Regression) both with gridsearch However prediction of 100000 target values is more than three times faster with SVR (Support Vector Regression) since it has learned a sparse model using only approx 13 of the 100 training datapoints as support vectors The next figure compares the time for fitting and prediction of KernelRidge and SVR for different sizes of the training set Fitting KernelRidge is faster than SVR (Support Vector Regression) for medium sized training sets less than 1000 samples however for larger training sets SVR (Support Vector Regression) scales better With regard to prediction time SVR (Support Vector Regression) is faster for all sizes of the training set because of the learned sparse solution Note that the degree of sparsity and thus the prediction time depends on the parameters ϵ and γ of the SVR $\epsilon = 0$ would correspond to a dense model

References

314 Support Vector Machines

Support vector machines SVMs are a set of supervised learning methods used for classification regression and outliers detection

31 Supervised learning 223

scikitlearn user guide Release 0213

The advantages of support vector machines are

- Effective in high dimensional spaces
- Still effective in cases where number of dimensions is greater than the number of samples
- Uses a subset of training points in the decision function called support vectors so it is also memory efficient
- Versatile different Kernel functions can be specified for the decision function Common kernels are provided but it is also possible to specify custom kernels

The disadvantages of support vector machines include

- If the number of features is much greater than the number of samples avoid overfitting in choosing Kernel functions and regularization term is crucial
- SVMs do not directly provide probability estimates these are calculated using an expensive fivefold cross validation see Scores and probabilities below

The support vector machines in scikitlearn support both dense `numpy.ndarray` and convertible to that by `numpy.asarray` and sparse any `scipy.sparse` sample vectors as input However to use an SVM (Support Vector Machine) to make predictions for sparse data it must have been fit on such data For optimal performance use `Cordered numpy.ndarray` dense or `scipy.sparse.csr_matrix` sparse with `dtype=float64`

Classification

`SVC` and `LinearSVC` are classes capable of performing multiclass classification on a dataset

scikitlearn user guide Release 0213

SVC (Support Vector Classification) andNuSVC are similar methods but accept slightly different sets of parameters and have different mathematical formulations see section Mathematical formulation On the other hand LinearSVC is another implementation of Support Vector Classification for the case of a linear kernel Note that LinearSVC does not accept keyword kernel as this is assumed to be linear It also lacks some of the members of SVC (Support Vector Classification) andNuSVC likesupport As other classifiers SVCNuSVC andLinearSVC take as input two arrays an array X of size nsamples nfeatures holding the training samples and an array y of class labels strings or integers size nsamples

```
from sklearn import svm
```

```
X = [[0, 0, 1, 1],
```

```
     [0, 1, 0, 1]]
```

```
y = [0, 1]
```

```
clf = svm.SVC(gamma=scale)
```

```
clf.fit(X, y)
```

```
SVCC10 cachesize=200 class_weight=None coef=0.00
```

```
decision_function_shape='ovr' degree=3 gamma=scale kernel='rbf'
```

```
max_iter=1 probability=False random_state=None shrinking=True
```

```
tol=0.001 verbose=False
```

After being fitted the model can then be used to predict new values

```
clf.predict([2, 2])
```

```
array([1])
```

SVMs decision function depends on some subset of the training data called the support vectors Some properties of

these support vectors can be found in members support_vectors support and n_support

```
clf.get_support_vectors()
```

```
clf.support_vectors_
```

```
array([[0, 0],
```

```
       [1, 1]])
```

```
clf.get_support_indices()
```

```
clf.support_
```

```
array([0, 1])
```

```
clf.get_n_support_vectors_for_each_class()
```

```
clf.n_support_
```

```
array([1, 1])
```

Multiclass classification

SVC (Support Vector Classification) andNuSVC implement the “oneagainstone” approach Knerr et al 1990 for multi class classifica

tion If nclass is the number of classes then nclass - 1 - 2 classifiers are con

structed and each one trains data from two classes To provide a consistent interface with other classifiers the

decision_function_shape option allows to monotonically transform the results of the “oneagainstone” classi

fiers to a decision function of shape nsamples nclasses

```
X = [[0, 1, 2, 3],
```

```
     [0, 1, 2, 3]]
```

```
Y = [0, 1, 2, 3]
```

```
clf = svm.SVC(gamma=scale, decision_function_shape='ovo')
```

```
clf.fit(X, Y)
```

```
SVCC10 cachesize=200 class_weight=None coef=0.00
```

```
decision_function_shape='ovo' degree=3 gamma=scale kernel='rbf'
```

```
max_iter=1 probability=False random_state=None shrinking=True
```

```
tol=0.001 verbose=False
```

```
dec_clf.decision_function(1)
```

```
dec_shape[1] = 4 # classes 4 32 6
```

```
6
```

31 Supervised learning 227

scikitlearn user guide Release 0213

clfdecisionfunctionsshape ovr

dec clfdecisionfunction1

decshape1 4 classes

4

On the other hand LinearSVC implements “onevsrest” multiclass strategy thus training nclass models If

there are only two classes only one model is trained

linclf svmLinearSVC

linclffitX Y

LinearSVCC10 classweightNone dualTrue fitinterceptTrue

interceptscaling1 losssquaredhinge maxiter1000

multiclassovr penaltyl2 randomstateNone tol00001

verbose0

dec linclfdecisionfunction1

decshape1

4

SeeMathematical formulation for a complete description of the decision function

Note that the LinearSVC also implements an alternative multiclass strategy the socalled multiclass SVM (Support Vector Machine) formu

lated by Crammer and Singer by using the option multiclasscrammersinger This method is consis

tent which is not true for onevsrest classification In practice onevsrest classification is usually preferred since

the results are mostly similar but the runtime is significantly less

For “onevsrest” LinearSVC the attributes coef andintercept have the shape nclass

nfeatures andnclass respectively Each row of the coefficients corresponds to one of the nclass

many “onevsrest” classifiers and similar for the intercepts in the order of the “one” class

In the case of “onevsone” SVC (Support Vector Classification) the layout of the attributes is a little more involved In the case of having

a linear kernel the attributes coef andintercept have the shape nclass nclass 1

2 nfeatures andnclass nclass 1 2 respectively This is similar to the layout for

LinearSVC described above with each row now corresponding to a binary classifier The order for classes 0 to n is

“0 vs 1” “0 vs 2” “0 vs n” “1 vs 2” “1 vs 3” “1 vs n” “n1 vs n”

The shape of dualcoef isnclass1 nSV with a somewhat hard to grasp layout The columns corre

spond to the support vectors involved in any of the nclass nclass 1 2 “onevsone” classifiers

Each of the support vectors is used in nclass 1 classifiers The nclass 1 entries in each row correspond

to the dual coefficients for these classifiers

This might be made more clear by an example

Consider a three class problem with class 0 having three support vectors $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$

and class 1 and 2 having two

support vectors $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$

and $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$

respectively For each support vector $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ there are two dual coefficients Let’s call

the coefficient of support vector $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$ in the classifier between classes $\begin{bmatrix} 0 \end{bmatrix}$ and $\begin{bmatrix} 1 \end{bmatrix}$ $\begin{bmatrix} 1 \end{bmatrix}$

$\begin{bmatrix} 1 \end{bmatrix}$ Thendualcoef looks like this

$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$

Coefficients for SVs of class 0

$\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 2 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Coefficients for SVs of class 1

$\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 2 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Coefficients for SVs of class 2

$\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$

228 Chapter 3 User Guide

Scores and probabilities

The `decisionfunction` method of `SVC` and `NuSVC` gives perclass scores for each sample or a single score per sample in the binary case. When the constructor option `probability` is set to `True`, class membership probability estimates from the methods `predict_proba` and `predict_log_proba` are enabled. In the binary case the probabilities are calibrated using Platt scaling (logistic regression on the SVM (Support Vector Machine)'s scores fit by an additional crossvalidation on the training data). In the multiclass case this is extended as per Wu et al 2004. Needless to say the crossvalidation involved in Platt scaling is an expensive operation for large datasets. In addition the probability estimates may be inconsistent with the scores in the sense that the "argmax" of the scores may not be the argmax of the probabilities. Eg in binary classification a sample may be labeled by `predict` as belonging to a class that has probability $\frac{1}{2}$ according to `predict_proba`. Platt's method is also known to have theoretical issues. If confidence scores are required but these do not have to be probabilities then it is advisable to set `probability=False` and use `decisionfunction` instead of `predict_proba`.

References

- Wu Lin and Weng "Probability estimates for multiclass classification by pairwise coupling" JMLR 5975 1005 2004
- Platt "Probabilistic outputs for SVMs and comparisons to regularized likelihood methods"

Unbalanced problems

In problems where it is desired to give more importance to certain classes or certain individual samples `class_weight` and `sample_weight` can be used. `SVC` (Support Vector Classification) but not `NuSVC` implement a keyword `class_weight` in the `fit` method. It's a dictionary of the form `classlabel: value` where `value` is a floating point number > 0 that sets the parameter `C` of class `classlabel` to `C*value`.

scikitlearn user guide Release 0213

SVCNuSVC SVRNuSVR andOneClassSVM implement also weights for individual samples in method fit through keyword sampleweight Similar to classweight these set the parameter Cfor the ith example to Csampleweighti

Examples

- Plot different SVM (Support Vector Machine) classifiers in the iris dataset
- SVM (Support Vector Machine) Maximum margin separating hyperplane
- SVM (Support Vector Machine) Separating hyperplane for unbalanced classes
- SVMANova SVM (Support Vector Machine) with univariate feature selection
- Nonlinear SVM (Support Vector Machine)
- SVM (Support Vector Machine) Weighted samples

Regression

The method of Support Vector Classification can be extended to solve regression problems This method is called Support Vector Regression

The model produced by support vector classification as described above depends only on a subset of the training data because the cost function for building the model does not care about training points that lie beyond the margin Analogously the model produced by Support Vector Regression depends only on a subset of the training data because the cost function for building the model ignores any training data close to the model prediction

There are three different implementations of Support Vector Regression SVRNuSVR andLinearSVR

LinearSVR provides a faster implementation than SVR (Support Vector Regression) but only considers linear kernels while NuSVR implements a slightly different formulation than SVR (Support Vector Regression) andLinearSVR See Implementation details for further details

As with classification classes the fit method will take as argument vectors X y only that in this case y is expected to have floating point values instead of integer values

```
from sklearn import svm
```

```
X 0 0 2 2
```

230 Chapter 3 User Guide

scikitlearn user guide Release 0213

```
y 05 25
clf svmSVR
clffitX y
SVRC10 cachesize200 coef000 degree3 epsilon01
gammaautodeprecated kernelrbf maxiter1 shrinkingTrue
tol0001 verboseFalse
clfpredict1 1
array15
```

Examples

- Support Vector Regression SVR (Support Vector Regression) using linear and nonlinear kernels

Density estimation novelty detection

The classOneClassSVM implements a OneClass SVM (Support Vector Machine) which is used in outlier detection

SeeNovelty and Outlier Detection for the description and usage of OneClassSVM

Complexity

Support Vector Machines are powerful tools but their compute and storage requirements increase rapidly with the number of training vectors The core of an SVM (Support Vector Machine) is a quadratic programming problem QP separating support vectors from the rest of the training data The QP solver used by this libsvmbased implementation scales between

n^2

and n^3

depending on how efficiently the libsvm cache is used in

practice dataset dependent If the data is very sparse n should be replaced by the average number of non zero features in a sample vector

Also note that for the linear case the algorithm used in LinearSVC by the liblinear implementation is much more efficient than its libsvmbased SVC (Support Vector Classification) counterpart and can scale almost linearly to millions of samples and/or features

Tips on Practical Use

- Avoiding data copy ForSVCSVRNuSVC andNuSVR if the data passed to certain methods is not Cordered contiguous and double precision it will be copied before calling the underlying C implementation You can check whether a given numpy array is Ccontiguous by inspecting its flags attribute ForLinearSVC andLogisticRegression any input passed as a numpy array will be copied and converted to the liblinear internal sparse data representation double precision floats and int32 indices of nonzero components If you want to fit a largescale linear classifier without copying a dense numpy Ccontiguous double precision array as input we suggest to use the SGDClassifier class instead The objective function can be configured to be almost the same as the LinearSVC model

- Kernel cache size ForSVCSVRNuSVC andNuSVR the size of the kernel cache has a strong impact on run times for larger problems If you have enough RAM available it is recommended to set cachesize to a higher value than the default of 200MB such as 500MB or 1000MB

- Setting C C is 1 by default and it's a reasonable default choice If you have a lot of noisy observations you should decrease it It corresponds to regularize more the estimation

LinearSVC andLinearSVR are less sensitive to Cwhen it becomes large and prediction results stop improving after a certain threshold Meanwhile larger C values will take more time to train sometimes up to 10 times longer as shown by Fan et al 2008

scikitlearn user guide Release 0213

- Support Vector Machine algorithms are not scale invariant so it is highly recommended to scale your data For example scale each attribute on the input vector X to 01 or 11 or standardize it to have mean 0 and variance 1 Note that the same scaling must be applied to the test vector to obtain meaningful results See section Preprocessing data for more details on scaling and normalization
- Parameter nu in NuSVC OneClassSVM NuSVR approximates the fraction of training errors and support vectors
- In SVC if data for classification are unbalanced eg many positive and few negative set class_weight=balanced and/or try different penalty parameters C
- Randomness of the underlying implementations The underlying implementations of SVC (Support Vector Classification) and NuSVC use a random number generator only to shuffle the data for probability estimation when probability is set to True This randomness can be controlled with the random_state parameter If probability is set to False these estimators are not random and random_state has no effect on the results The underlying OneClassSVM implementation is similar to the ones of SVC (Support Vector Classification) and NuSVC As no probability estimation is provided for OneClassSVM it is not random The underlying LinearSVC implementation uses a random number generator to select features when fitting the model with a dual coordinate descent ie when dual is set to True It is thus not uncommon to have slightly different results for the same input data If that happens try with a smaller tol parameter This randomness can also be controlled with the random_state parameter When dual is set to False the underlying implementation of LinearSVC is not random and random_state has no effect on the results
- Using L1 penalization as provided by LinearSVC loss_l2 penalty_l1 dual=False yields a sparse solution ie only a subset of feature weights is different from zero and contribute to the decision function Increasing C yields a more complex model more features are selected The C value that yields a “null” model all weights equal to zero can be calculated using l1_minc

References

- Fan RongEn et al “LIBLINEAR A library for large linear classification” Journal of machine learning research 9 Aug 2008 1871-1874

Kernel functions

The kernel function can be any of the following

- linear(γ)
- polynomial $\langle \mathbf{x} \cdot \mathbf{x}' \rangle^d$ is specified by keyword degree d by coef0
- rbf $\exp(-\gamma \| \mathbf{x} - \mathbf{x}' \|^2)$ is specified by keyword gamma must be greater than 0
- sigmoid $\tanh(\gamma \langle \mathbf{x} \cdot \mathbf{x}' \rangle)$ where γ is specified by coef0

Different kernels are specified by keyword kernel at initialization

```
linearsvc svmSVCKernel(linear)
linearsvc Kernel
linear
rbfsvc svmSVCKernel(rbf)
rbfsvc Kernel
rbf
```

Custom Kernels

You can define your own kernels by either giving the kernel as a python function or by precomputing the Gram matrix
Classifiers with custom kernels behave the same way as any other classifiers except that

- Fieldsupportvectors is now empty only indices of support vectors are stored in support
- A reference and not a copy of the first argument in the fit method is stored for future reference If that array changes between the use of fit andpredict you will have unexpected results

Using Python functions as kernels

You can also use your own defined kernels by passing a function to the keyword kernel in the constructor

Your kernel must take as arguments two matrices of shape nsamples1 nfeatures nsamples2
nfeatures and return a kernel matrix of shape nsamples1 nsamples2

The following code defines a linear kernel and creates a classifier instance that will use that kernel

```
import numpy as np
from sklearn import svm
def mykernel(X, Y):
    return np.dot(X, Y.T)

clf = svm.SVC(kernel=mykernel)
```

Examples

- SVM (Support Vector Machine) with custom kernel

Using the Gram matrix

Setkernelprecomputed and pass the Gram matrix instead of X in the fit method At the moment the kernel values between alltraining vectors and the test vectors must be provided

```
import numpy as np
from sklearn import svm
X = np.array([0, 0, 1, 1])
y = [0, 1]
clf = svm.SVC(kernel=precomputed)
# linear kernel computation
gram = np.dot(X, X.T)
clf.fit(gram, y)
SVCC10 cachesize200 classweightNone coef000
decisionfunctionshapeovr degree3 gammaautodeprecated
kernelprecomputed maxiter1 probabilityFalse
randomstateNone shrinkingTrue tol0001 verboseFalse
# predict on training examples
clf.predict(gram)
array([0, 1])
```

scikitlearn user guide Release 0213

Parameters of the RBF (Radial Basis Function) Kernel

When training an SVM (Support Vector Machine) with the Radial Basis Function RBF (Radial Basis Function) kernel two parameters must be considered: C and γ . The parameter C common to all SVM (Support Vector Machine) kernels trades off misclassification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth while a high C aims at classifying all training examples correctly. γ defines how much influence a single training example has. The larger γ is, the closer other examples must be to be affected.

Proper choice of C and γ is critical to the SVM (Support Vector Machine)'s performance. One is advised to use `sklearn.model_selection.GridSearchCV` with C and γ spaced exponentially far apart to choose good values.

Examples

- RBF (Radial Basis Function) SVM (Support Vector Machine) parameters

Mathematical formulation

A support vector machine constructs a hyperplane or set of hyperplanes in a high or infinite dimensional space which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data points of any class, so-called functional margin. Since in general the larger the margin the lower the generalization error of the classifier.

234 Chapter 3 User Guide

SVC (Support Vector Classification)

Given training vectors $x_i \in \mathbb{R}^n$ in two classes and a vector $\gamma \in [1, -1]$ SVC (Support Vector Classification) solves the following primal problem

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{i=1}^n \xi_i \\ \text{subject to} \quad & \gamma x_i^T w + b \geq 1 - \xi_i \\ & \xi_i \geq 0 \quad \forall i \end{aligned}$$

Its dual is

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{i=1}^n \alpha_i^2 \\ \text{subject to} \quad & \alpha_i \geq 0 \\ & 0 \leq \alpha_i \leq \gamma_i \quad \forall i \end{aligned}$$

where γ is the vector of all ones, γ_i is the upper bound, α is an n -by- n positive semidefinite matrix $\alpha_{ij} = \gamma_i \gamma_j K(x_i, x_j)$ where $K(x_i, x_j)$ is the kernel. Here training vectors are implicitly mapped into a higher maybe infinite dimensional space by the function ϕ .

The decision function is

$$\text{sgn} \left(\sum_{i=1}^n \alpha_i \gamma_i \phi(x_i)^T \phi(x) + b \right)$$

Note While SVM (Support Vector Machine) models derived from libsvm and liblinear use C as regularization parameter most other estimators use α . The exact equivalence between the amount of regularization of two models depends on the exact objective function optimized by the model. For example when the estimator used is `sklearn.linear_model.RidgeRegression` the relation between them is given as $\alpha = 1/C$.

This parameters can be accessed through the members `dual_coef` which holds the product $\sum \alpha_i \gamma_i \phi(x_i)$, `support_vectors` which holds the support vectors and `intercept` which holds the independent term b .

References

- “Automatic Capacity Tuning of Very Large VCdimension Classifiers” I Guyon B Boser V Vapnik Advances in neural information processing 1993
- “Supportvector networks” C Cortes V Vapnik Machine Learning 20 273297 1995

NuSVC

We introduce a new parameter ν which controls the number of support vectors and training errors. The parameter $\nu \in [0, 1]$ is an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. It can be shown that the ν -SVC formulation is a reparameterization of the C -SVC and therefore mathematically equivalent.

SVR (Support Vector Regression)

Given training vectors $x_i \in \mathbb{R}^n$ and a vector $y \in \mathbb{R}$ SVR solves the following primal problem

$$\begin{aligned} \min_{\alpha, \xi} \quad & \frac{1}{2} \sum_{i=1}^n \xi_i^2 \\ \text{subject to} \quad & y_i - \sum_{j=1}^n \alpha_j x_{ij} - \xi_i \leq 0 \\ & \xi_i \geq 0 \end{aligned}$$

Its dual is

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^n \xi_i^2 \\ \text{subject to} \quad & 0 \leq \alpha_i \leq 1 \end{aligned}$$

where $\mathbf{1}$ is the vector of all ones ξ is the upper bound \mathbf{K} is an $n \times n$ positive semidefinite matrix $\mathbf{K}_{ij} = \kappa(x_i, x_j)$ is the kernel Here training vectors are implicitly mapped into a higher maybe infinite dimensional space by the function ϕ

The decision function is

$$\hat{y} = \sum_{i=1}^n \alpha_i (y_i - \xi_i) + \xi_0$$

These parameters can be accessed through the members `dualcoef` which holds the difference $y_i - \xi_i$ `supportvectors` which holds the support vectors and `intercept` which holds the independent term ξ_0

References

- “A Tutorial on Support Vector Regression” Alex J Smola Bernhard Schölkopf Statistics and Computing archive Volume 14 Issue 3 August 2004 p 199222

Implementation details

Internally we use `libsvm` and `liblinear` to handle all computations These libraries are wrapped using C and Cython

References

For a description of the implementation and details of the algorithms used please refer to

- LIBSVM A Library for Support Vector Machines
- LIBLINEAR – A Library for Large Linear Classification

315 Stochastic Gradient Descent

Stochastic Gradient Descent SGD is a simple yet very efficient approach to discriminative learning of linear classifiers under convex loss functions such as linear Support Vector Machines and Logistic Regression Even though

scikitlearn user guide Release 0213

SGD has been around in the machine learning community for a long time it has received a considerable amount of attention just recently in the context of largescale learning
SGD has been successfully applied to largescale and sparse machine learning problems often encountered in text classification and natural language processing Given that the data is sparse the classifiers in this module easily scale to problems with more than 105 training examples and more than 105 features

The advantages of Stochastic Gradient Descent are

- Efficiency
- Ease of implementation lots of opportunities for code tuning

The disadvantages of Stochastic Gradient Descent include

- SGD requires a number of hyperparameters such as the regularization parameter and the number of iterations
- SGD is sensitive to feature scaling

Classification

Warning Make sure you permute shuffle your training data before fitting the model or use shuffleTrue

to shuffle after each iteration

The classSGDClassifier implements a plain stochastic gradient descent learning routine which supports different loss functions and penalties for classification

As other classifiers SGD has to be fitted with two arrays an array X of size nsamples nfeatures holding the training samples and an array Y of size nsamples holding the target values class labels for the training samples

from sklearnlinearmodel import SGDClassifier

X 0 0 1 1

31 Supervised learning 237

scikitlearn user guide Release 0213

```
y 0 1
clf = SGDClassifier(loss=shinge, penalty=l2, max_iter=5
clf.fit(X, y)
SGDClassifier(alpha=0.0001, average=False, class_weight=None
early_stopping=False, epsilon=0.1, eta=0.00, fit_intercept=True
l1_ratio=0.15, learning_rate=optimal, loss=shinge, max_iter=5
n_iter_no_change=5, n_jobs=None, penalty=l2, power_t=0.5
random_state=None, shuffle=True, tol=0.0001
validation_fraction=0.1, verbose=0, warm_start=False
After being fitted the model can then be used to predict new values
clf.predict(2, 2)
array(1)
```

SGD fits a linear model to the training data. The member coef holds the model parameters

```
clf.coef
array(99, 99)
Member.intercept holds the intercept aka offset or bias
clf.intercept
array(99)
```

Whether or not the model should use an intercept ie a biased hyperplane is controlled by the parameter fit_intercept

To get the signed distance to the hyperplane use SGDClassifier.decision_function

```
clf.decision_function(2, 2)
array(296)
The concrete loss function can be set via the loss parameter. SGDClassifier supports the following loss functions
```

- loss=shinge: softmargin linear Support Vector Machine
- loss=modifiedhuber: smoothed hinge loss
- loss=log: logistic regression
- and all regression losses below

The first two loss functions are lazy: they only update the model parameters if an example violates the margin constraint which makes training very efficient and may result in sparser models even when L2 penalty is used. Using loss=log or loss=modifiedhuber enables the predict_proba method which gives a vector

```
of probability estimates (per sample)
clf = SGDClassifier(loss=log, max_iter=5)
clf.fit(X, y)
clf.predict_proba(1, 1)
array(0.00, 0.99)
```

The concrete penalty can be set via the penalty parameter. SGD supports the following penalties

- penalty=l2: L2 norm penalty on coef
- penalty=l1: L1 norm penalty on coef

- penaltyelasticnet Convex combination of L2 and L1 1 l1ratio L2 l1ratio L1

The default setting is penaltyl2 The L1 penalty leads to sparse solutions driving most coefficients to zero The Elastic Net solves some deficiencies of the L1 penalty in the presence of highly correlated attributes The parameterl1ratio controls the convex combination of L1 and L2 penalty

SGDClassifier supports multiclass classification by combining multiple binary classifiers in a “one versus all” OV A scheme For each of the K classes a binary classifier is learned that discriminates between that and all other $K-1$ classes At testing time we compute the confidence score ie the signed distances to the hyperplane for each classifier and choose the class with the highest confidence The Figure below illustrates the OV A approach on the iris dataset The dashed lines represent the three OV A classifiers the background colors show the decision surface induced by the three classifiers

In the case of multiclass classification coef is a twodimensional array of shapenclasses nfeatures andintercept is a onedimensional array of shapenclasses The ith row of coef holds the weight vector of the OV A classifier for the ith class classes are indexed in ascending order see at tributeclasses Note that in principle since they allow to create a probability model losslog and lossmodifiedhuber are more suitable for onevsall classification SGDClassifier supports both weighted classes and weighted instances via the fit parameters classweight andsampleweight See the examples below and the docstring of SGDClassifierfit for further information

Examples

- SGD Maximum margin separating hyperplane
 - Plot multiclass SGD on the iris dataset
 - SGD Weighted samples
 - Comparing various online solvers
- 31 Supervised learning 239

scikitlearn user guide Release 0213

•SVM (Support Vector Machine) Separating hyperplane for unbalanced classes See theNote

SGDClassifier supports averaged SGD ASGD Averaging can be enabled by setting averageTrue

ASGD works by averaging the coefficients of the plain SGD over each iteration over a sample When using ASGD

the learning rate can be larger and even constant leading on some datasets to a speed up in training time

For classification with a logistic loss another variant of SGD with an averaging strategy is available with Stochastic

Average Gradient SAG algorithm available as a solver in LogisticRegression

Regression

The classSGDRegressor implements a plain stochastic gradient descent learning routine which supports different

loss functions and penalties to fit linear regression models SGDRegressor is well suited for regression prob

lems with a large number of training samples 10000 for other problems we recommend Ridge Lasso or

ElasticNet

The concrete loss function can be set via the loss parameterSGDRegressor supports the following loss functions

•losssquaredloss Ordinary least squares

•losshuber Huber loss for robust regression

•lossepsiloninsensitive linear Support Vector Regression

The Huber and epsiloninsensitive loss functions can be used for robust regression The width of the insensitive region

has to be specified via the parameter epsilon This parameter depends on the scale of the target variables

SGDRegressor supports averaged SGD as SGDClassifier Averaging can be enabled by setting

averageTrue

For regression with a squared loss and a l2 penalty another variant of SGD with an averaging strategy is available with

Stochastic Average Gradient SAG algorithm available as a solver in Ridge

Stochastic Gradient Descent for sparse data

Note The sparse implementation produces slightly different results than the dense implementation due to a shrunk

learning rate for the intercept

There is builtin support for sparse data given in any matrix in a format supported by scipysparse For maximum

efficiency however use the CSR (Compressed Sparse Row) matrix format as defined in scipysparsecsrmatrix

Examples

•Classification of text documents using sparse features

Complexity

The major advantage of SGD is its efficiency which is basically linear in the number of training examples If X is a

matrix of size n p training has a cost of kp where k is the number of iterations epochs and p is the average

number of nonzero attributes per sample

Recent theoretical results however show that the runtime to get some desired optimization accuracy does not increase

as the training set size increases

240 Chapter 3 User Guide

Stopping criterion

The classes SGDClassifier andSGDRegressor provide two criteria to stop the algorithm when a given level of convergence is reached

- WithearlystoppingTrue the input data is split into a training set and a validation set The model is then fitted on the training set and the stopping criterion is based on the prediction score computed on the validation set The size of the validation set can be changed with the parameter validationfraction
- WithearlystoppingFalse the model is fitted on the entire input data and the stopping criterion is based on the objective function computed on the input data

In both cases the criterion is evaluated once by epoch and the algorithm stops when the criterion does not improve niternochange times in a row The improvement is evaluated with a tolerance tol and the algorithm stops in any case after a maximum number of iteration maxiter

Tips on Practical Use

- Stochastic Gradient Descent is sensitive to feature scaling so it is highly recommended to scale your data For example scale each attribute on the input vector X to 01 or 11 or standardize it to have mean 0 and variance 1 Note that the same scaling must be applied to the test vector to obtain meaningful results This can be easily done using StandardScaler

from sklearnpreprocessing import StandardScaler
scaler = StandardScaler

scaler.fit(Xtrain) Dont cheat fit only on training data

Xtrain = scaler.transform(Xtrain)

Xtest = scaler.transform(Xtest) apply same transformation to test data

If your attributes have an intrinsic scale eg word frequencies or indicator features scaling is not needed

- Finding a reasonable regularization term λ is best done using GridSearchCV usually in the range 10^{-5} to 10^{-1}

- Empirically we found that SGD converges after observing approx 10^6 training samples Thus a reasonable first guess for the number of iterations is $\max(\maxiter, \lceil 10^6 / n \rceil)$ where n is the size of the training set

- If you apply SGD to features extracted using PCA (Principal Component Analysis) we found that it is often wise to scale the feature values by some constant c such that the average L2 norm of the training data equals one

- We found that Averaged SGD works best with a larger number of features and a higher η

References

- “Efficient BackProp” Y. LeCun, L. Bottou, G. Orr, K. Müller. In Neural Networks Tricks of the Trade 1998

Mathematical formulation

Given a set of training examples $\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$ our goal is to learn a linear scoring function $f(x) = w^T x + b$ with model parameters $w \in \mathbb{R}^d$ and intercept $b \in \mathbb{R}$. In order to make predictions we simply look at the sign of $f(x)$. A common choice to find the model parameters is by minimizing the regularized training error given by

$$J(w, b) = \frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i (w^T x_i + b)) + \frac{\lambda}{2} \|w\|^2$$

where ℓ is a loss function that measures model misfit and λ is a regularization term aka penalty that penalizes model complexity. $\lambda \geq 0$ is a nonnegative hyperparameter.

Different choices for ℓ entail different classifiers such as

- Hinge softmargin Support Vector Machines
- Log Logistic Regression
- LeastSquares Ridge Regression
- EpsilonInsensitive softmargin Support Vector Regression

All of the above loss functions can be regarded as an upper bound on the misclassification error. Zero-one loss as shown in the Figure below.

Popular choices for the regularization term λ include

- L2 norm $\lambda \sum_i w_i^2$

$\lambda \sum_i |w_i|$

- L1 norm $\lambda \sum_i |w_i|$

$\lambda \sum_i |w_i|$ which leads to sparse solutions

- Elastic Net $\lambda (\alpha \sum_i |w_i| + (1-\alpha) \sum_i w_i^2)$

$\lambda \sum_i |w_i|$

$\lambda \sum_i w_i^2$

$\lambda \sum_i |w_i|$

$\lambda \sum_i |w_i|$

$\lambda \sum_i |w_i|$ a convex combination of L2 and L1 where α is given

by $\alpha = \frac{\lambda_1}{\lambda_1 + \lambda_2}$

The Figure below shows the contours of the different regularization terms in the parameter space when $\lambda \geq 0$.

SGD

Stochastic gradient descent is an optimization method for unconstrained optimization problems. In contrast to batch

gradient descent, SGD approximates the true gradient of $J(w)$ by considering a single training example at a time.

The class `SGDClassifier` implements a first-order SGD learning routine. The algorithm iterates over the training

examples and for each example updates the model parameters according to the update rule given by

$$\theta \leftarrow \theta - \eta \nabla_{\theta} L(\theta; x, y)$$

where η is the learning rate which controls the stepsize in the parameter space The intercept θ_0 is updated similarly but without regularization

The learning rate η can be either constant or gradually decaying For classification the default learning rate schedule `learningrateoptimal` is given by

$$\eta = \frac{1}{\sqrt{n \cdot t}}$$

where t is the time step there are a total of `nsamples` niter time steps η is determined based on a heuristic proposed by Léon Bottou such that the expected initial updates are comparable with the expected size of the weights this assuming that the norm of the training samples is approx 1 The exact definition can be found in `initt` in `BaseSGD`

For regression the default learning rate schedule is inverse scaling `learningrateinvscaling` given by

$$\eta = \frac{\eta_0}{1 + \text{power} \cdot t}$$

where η_0 and `power` are hyperparameters chosen by the user via `eta0` and `power` resp

For a constant learning rate use `learningrateconstant` and `useeta0` to specify the learning rate

For an adaptively decreasing learning rate use `learningrateadaptive` and `useeta0` to specify the start

ing learning rate When the stopping criterion is reached the learning rate is divided by 5 and the algorithm does not stop The algorithm stops when the learning rate goes below `1e6`

The model parameters can be accessed through the members `coef` and `intercept`

- `Membercoef` holds the weights θ
- `Memberintercept` holds θ_0

References

- “Solving large scale linear prediction problems using stochastic gradient descent algorithms” T Zhang In Proceedings of ICML ‘04
- “Regularization and variable selection via the elastic net” H Zou T Hastie Journal of the Royal Statistical Society Series B 67 2 301320
- “Towards Optimal One Pass Large Scale Learning with Averaged Stochastic Gradient Descent” Xu Wei

Implementation details

The implementation of SGD is influenced by the Stochastic Gradient SVM (Support Vector Machine) of Léon Bottou Similar to `SvmSGD` the weight vector is represented as the product of a scalar and a vector which allows an efficient weight update in the case of L2 regularization In the case of sparse feature vectors the intercept is updated with a smaller learning rate multiplied by 0.01 to account for the fact that it is updated more frequently Training examples are picked up sequentially and the learning rate is lowered after each observed example We adopted the learning rate schedule from ShalevShwartz et al 2007 For multiclass classification a “one versus all” approach is used We use the truncated gradient algorithm proposed by Tsuruoka et al 2009 for L1 regularization and the Elastic Net The code is written in Cython

References

- “Stochastic Gradient Descent” L Bottou Website 2010
- “The Tradeoffs of Large Scale Machine Learning” L Bottou Website 2011
- “Pegasos Primal estimated subgradient solver for svm” S ShalevShwartz Y Singer N Srebro In Proceedings of ICML ‘07
- “Stochastic gradient descent training for l1regularized loglinear models with cumulative penalty” Y Tsuruoka J Tsujii S Ananiadou In Proceedings of the AFNLPACL ‘09

316 Nearest Neighbors

sklearnneighbors provides functionality for unsupervised and supervised neighborsbased learning methods

Unsupervised nearest neighbors is the foundation of many other learning methods notably manifold learning and spectral clustering Supervised neighborsbased learning comes in two flavors classification for data with discrete labels and regression for data with continuous labels

The principle behind nearest neighbor methods is to find a predefined number of training samples closest in distance to the new point and predict the label from these The number of samples can be a userdefined constant knearest neighbor learning or vary based on the local density of points radiusbased neighbor learning The distance can in general be any metric measure standard Euclidean distance is the most common choice Neighborsbased methods are known as nongeneralizing machine learning methods since they simply “remember” all of its training data possibly transformed into a fast indexing structure such as a Ball Tree orKD Tree

Despite its simplicity nearest neighbors has been successful in a large number of classification and regression problems including handwritten digits and satellite image scenes Being a nonparametric method it is often successful in classification situations where the decision boundary is very irregular

The classes in sklearnneighbors can handle either NumPy arrays or scipysparse matrices as input For dense matrices a large number of possible distance metrics are supported For sparse matrices arbitrary Minkowski metrics are supported for searches

There are many learning routines which rely on nearest neighbors at their core One example is kernel density estimation discussed in the density estimation section

Unsupervised Nearest Neighbors

NearestNeighbors implements unsupervised nearest neighbors learning It acts as a uniform interface to three different nearest neighbors algorithms BallTree KDTree and a brute force algorithm based on routines in sklearnmetricspairwise The choice of neighbors search algorithm is controlled through the keyword algorithm which must be one of auto balltree kdtree brute When the default value auto is passed the algorithm attempts to determine the best approach from the training data For a discussion of the strengths and weaknesses of each option see Nearest Neighbor Algorithms

Warning Regarding the Nearest Neighbors algorithms if two neighbors x_1 and x_2 have identical distances but different labels the result will depend on the ordering of the training data

31 Supervised learning 245

Finding the Nearest Neighbors

For the simple task of finding the nearest neighbors between two sets of data the unsupervised algorithms within sklearnneighbors can be used

```
from sklearnneighbors import NearestNeighbors
import numpy as np
X nparray1 1 2 1 3 2 1 1 2 1 3 2
nbrs NearestNeighborsnneighbors2 algorithmballtreefitX
distances indices nbrskneighborsX
```

```
indices
array0 1
1 0
2 1
3 4
4 3
5 4
distances
array0 1
0 1
0 141421356
0 1
0 1
0 141421356
```

Because the query set matches the training set the nearest neighbor of each point is the point itself at a distance of zero

It is also possible to efficiently produce a sparse graph showing the connections between neighboring points

```
nbrskneighborsgraphXtoarray
array1 1 0 0 0 0
1 1 0 0 0 0
0 1 1 0 0 0
0 0 0 1 1 0
0 0 0 1 1 0
0 0 0 0 1 1
```

The dataset is structured such that points nearby in index order are nearby in parameter space leading to an approximately blockdiagonal matrix of Knearest neighbors Such a sparse graph is useful in a variety of circumstances which make use of spatial relationships between points for unsupervised learning in particular seeksklearnmanifoldIsomap sklearnmanifoldLocallyLinearEmbedding andsklearn

clusterSpectralClustering
KDTree and BallTree Classes

Alternatively one can use the KDTree orBallTree classes directly to find nearest neighbors This is the functionality wrapped by the NearestNeighbors class used above The Ball Tree and KD Tree have the same interface we'll show an example of using the KD Tree here

```
from sklearnneighbors import KDTree
import numpy as np
X nparray1 1 2 1 3 2 1 1 2 1 3 2
kdt KDTreeX leafsize30 metric=euclidean
kdtqueryX k2 returndistance=False
array0 1
```

1 0
2 1
3 4
4 3
5 4

Refer to the KDTree andBallTree class documentation for more information on the options available for nearest neighbors searches including specification of query strategies distance metrics etc For a list of available metrics see the documentation of the DistanceMetric class

Nearest Neighbors Classification

Neighborsbased classification is a type of instancebased learning ornongeneralizing learning it does not attempt to construct a general internal model but simply stores instances of the training data Classification is computed from a simple majority vote of the nearest neighbors of each point a query point is assigned the data class which has the most representatives within the nearest neighbors of the point

scikitlearn implements two different nearest neighbors classifiers KNeighborsClassifier implements learning based on the k nearest neighbors of each query point where k is an integer value specified by the user

RadiusNeighborsClassifier implements learning based on the number of neighbors within a fixed radius ϵ of each training point where ϵ is a floatingpoint value specified by the user

The k neighbors classification in KNeighborsClassifier is the most commonly used technique The optimal choice of the value k is highly datadependent in general a larger k suppresses the effects of noise but makes the classification boundaries less distinct

In cases where the data is not uniformly sampled radiusbased neighbors classification in

RadiusNeighborsClassifier can be a better choice The user specifies a fixed radius ϵ such that

points in sparser neighborhoods use fewer nearest neighbors for the classification For highdimensional parameter spaces this method becomes less effective due to the socalled “curse of dimensionality”

The basic nearest neighbors classification uses uniform weights that is the value assigned to a query point is computed from a simple majority vote of the nearest neighbors Under some circumstances it is better to weight the neighbors such that nearer neighbors contribute more to the fit This can be accomplished through the weights keyword The default value weights uniform assigns uniform weights to each neighbor weights distance assigns weights proportional to the inverse of the distance from the query point Alternatively a userdefined function of the distance can be supplied to compute the weights

Examples

•Nearest Neighbors Classification an example of classification using nearest neighbors

Nearest Neighbors Regression

Neighborsbased regression can be used in cases where the data labels are continuous rather than discrete variables

The label assigned to a query point is computed based on the mean of the labels of its nearest neighbors

scikitlearn implements two different neighbors regressors KNeighborsRegressor implements learning

based on the k nearest neighbors of each query point where k is an integer value specified by the user

RadiusNeighborsRegressor implements learning based on the neighbors within a fixed radius ϵ of the query

point where ϵ is a floatingpoint value specified by the user

The basic nearest neighbors regression uses uniform weights that is each point in the local neighborhood contributes

uniformly to the classification of a query point Under some circumstances it can be advantageous to weight points

such that nearby points contribute more to the regression than faraway points This can be accomplished through the

weights keyword The default value weights uniform assigns equal weights to all points weights

distance assigns weights proportional to the inverse of the distance from the query point Alternatively a

userdefined function of the distance can be supplied which will be used to compute the weights

The use of multioutput nearest neighbors for regression is demonstrated in Face completion with a multioutput

estimators In this example the inputs X are the pixels of the upper half of faces and the outputs Y are the pixels of

the lower half of those faces

Examples

•Nearest Neighbors regression an example of regression using nearest neighbors

•Face completion with a multioutput estimators an example of multioutput regression using nearest neighbors

Nearest Neighbor Algorithms

Brute Force

Fast computation of nearest neighbors is an active area of research in machine learning The most naive neighbor search implementation involves the brute force computation of distances between all pairs of points in the dataset for n samples in d dimensions this approach scales as $O(n^2d)$ Efficient brute force neighbors searches can be very competitive for small data samples However as the number of samples n grows the brute force approach quickly becomes infeasible In the classes within `sklearn.neighbors` brute force neighbors searches are specified using the keyword algorithm `brute` and are computed using the routines available in `sklearn.metrics`

pairwise

KD Tree

To address the computational inefficiencies of the brute force approach a variety of tree based data structures have been invented In general these structures attempt to reduce the required number of distance calculations by efficiently encoding aggregate distance information for the sample The basic idea is that if point p is very distant from point q and point r is very close to point q then we know that points p and r are very distant without having to explicitly calculate their distance In this way the computational cost of a nearest neighbors search can be reduced to $O(n \log n)$ or better This is a significant improvement over brute force for large n

An early approach to taking advantage of this aggregate information was the KD tree data structure short for K dimensional tree which generalizes two dimensional Quad trees and 3 dimensional Oct trees to an arbitrary number of dimensions The KD tree is a binary tree structure which recursively partitions the parameter space along the data axes dividing it into nested orthotropic regions into which data points are filed The construction of a KD tree is very fast because partitioning is performed only along the data axes no d dimensional distances need to be computed Once constructed the nearest neighbor of a query point can be determined with only $O(\log n)$ distance computations Though the KD tree approach is very fast for low dimensional $d \leq 20$ neighbors searches it becomes inefficient as n grows very large this is one manifestation of the so called “curse of dimensionality” In scikitlearn KD tree neighbors searches are specified using the keyword algorithm `kdtree` and are computed using the class

KDTree

References

- “Multidimensional binary search trees used for associative searching” Bentley JL Communications of the ACM 1975

Ball Tree

To address the inefficiencies of KD Trees in higher dimensions the ball tree data structure was developed Where KD trees partition data along Cartesian axes ball trees partition data in a series of nesting hyperspheres This makes tree construction more costly than that of the KD tree but results in a data structure which can be very efficient on highly structured data even in very high dimensions

A ball tree recursively divides the data into nodes defined by a centroid c and radius r such that each point in the node lies within the hypersphere defined by c and r The number of candidate points for a neighbor search is reduced

scikitlearn user guide Release 0213
through use of the triangle inequality
 $d(x, c) \leq d(x, y) + d(y, c)$

With this setup a single distance calculation between a test point and the centroid is sufficient to determine a lower and upper bound on the distance to all points within the node Because of the spherical geometry of the ball tree nodes it can outperform a KDtree in high dimensions though the actual performance is highly dependent on the structure of the training data In scikitlearn balltreebased neighbors searches are specified using the keyword algorithm 'balltree' and are computed using the class sklearn.neighbors.BallTree Alternatively the user can work with the BallTree class directly

References
• "Five balltree construction algorithms" Omohundro SM International Computer Science Institute Technical Report 1989

Choice of Nearest Neighbors Algorithm

The optimal algorithm for a given dataset is a complicated choice and depends on a number of factors

- number of samples n and dimensionality d

-Brute force query time grows as n^2

-Ball tree query time grows as approximately $n \log n$

-KD tree query time changes with d in a way that is difficult to precisely characterise For small d less than 20 or so the cost is approximately $n \log n$ and the KD tree query can be very efficient For larger d the cost increases to nearly n^2 and the overhead due to the tree structure can lead to queries which are slower than brute force

For small data sets d less than 30 or so $\log d$ is comparable to d and brute force algorithms can be more efficient than a treebased approach Both KDTree and BallTree address this through providing a leaf size parameter this controls the number of samples at which a query switches to brute force This allows both algorithms to approach the efficiency of a brute force computation for small d

- data structure intrinsic dimensionality of the data and/or sparsity of the data Intrinsic dimensionality refers to the dimension $k \leq d$ of a manifold on which the data lies which can be linearly or nonlinearly embedded in the parameter space Sparsity refers to the degree to which the data fills the parameter space this is to be distinguished from the concept as used in "sparse" matrices The data matrix may have no zero entries but the structure can still be "sparse" in this sense

-Brute force query time is unchanged by data structure

-Ball tree and KD tree query times can be greatly influenced by data structure In general sparser data with a smaller intrinsic dimensionality leads to faster query times Because the KD tree internal representation is aligned with the parameter axes it will not generally show as much improvement as ball tree for arbitrarily structured data

Datasets used in machine learning tend to be very structured and are very well suited for treebased queries

- number of neighbors k requested for a query point

-Brute force query time is largely unaffected by the value of k

-Ball tree and KD tree query time will become slower as k increases This is due to two effects first a larger k leads to the necessity to search a larger portion of the parameter space Second using $k = 1$ requires internal queueing of results as the tree is traversed

As n becomes large compared to k the ability to prune branches in a treebased query is reduced In this situation Brute force queries can be more efficient

- number of query points Both the ball tree and the KD Tree require a construction phase The cost of this construction becomes negligible when amortized over many queries If only a small number of queries will be performed however the construction can make up a significant fraction of the total cost If very few query points will be required brute force is better than a treebased method

Currently algorithm `auto` selects `brute` if $n \leq 2$ the input data is sparse or `effectivemetric` isn't in the `VALIDMETRICS` list for either `kdtree` or `balltree` Otherwise it selects the first out of `kdtree` and `balltree` that has `effectivemetric` in its `VALIDMETRICS` list This choice is based on the assumption that the number of query points is at least the same order as the number of training points and that `leafsize` is close to its default value of 30

Effect of `leafsize`

As noted above for small sample sizes a brute force search can be more efficient than a treebased query This fact is accounted for in the ball tree and KD tree by internally switching to brute force searches within leaf nodes The level of this switch can be specified with the parameter `leafsize` This parameter choice has many effects

construction time A larger `leafsize` leads to a faster tree construction time because fewer nodes need to be created

query time Both a large or small `leafsize` can lead to suboptimal query cost For `leafsize` approaching 1 the overhead involved in traversing nodes can significantly slow query times For `leafsize` approaching the size of the training set queries become essentially brute force A good compromise between these is `leafsize = 30` the default value of the parameter

memory As `leafsize` increases the memory required to store a tree structure decreases This is especially important in the case of ball tree which stores a k -dimensional centroid for each node The required storage space for `BallTree` is approximately $1 / \text{leafsize}$ times the size of the training set

`leafsize` is not referenced for brute force queries

Nearest Centroid Classifier

The `NearestCentroid` classifier is a simple algorithm that represents each class by the centroid of its members In effect this makes it similar to the label updating phase of the `sklearnKMeans` algorithm It also has no parameters to choose making it a good baseline classifier It does however suffer on nonconvex classes as well as when classes have drastically different variances as equal variance in all dimensions is assumed See `Linear Discriminant Analysis` `sklearn discriminant analysis LinearDiscriminantAnalysis` and `Quadratic Discriminant Analysis` `sklearn discriminant analysis QuadraticDiscriminantAnalysis` for more complex methods that do not make this assumption Usage of the default `NearestCentroid` is simple

```
from sklearn.neighbors.nearestcentroid import NearestCentroid
import numpy as np
X = np.array([1 2 1 3 2 1 1 2 1 3 2])
y = np.array([1 1 1 2 2 2])
clf = NearestCentroid
clf.fit(X, y)
NearestCentroid(metric=euclidean shrinkthreshold=None)
print(clf.predict([0.8, 1.1]))
```

Nearest Shrunk Centroid

TheNearestCentroid classifier has a shrinkthreshold parameter which implements the nearest shrunken centroid classifier In effect the value of each feature for each centroid is divided by the withinclass variance of that feature The feature values are then reduced by shrinkthreshold Most notably if a particular feature value crosses zero it is set to zero In effect this removes the feature from affecting the classification This is useful for example for removing noisy features

In the example below using a small shrink threshold increases the accuracy of the model from 081 to 082

Examples

- Nearest Centroid Classification an example of classification using nearest centroid with different shrink thresholds

Neighborhood Components Analysis

Neighborhood Components Analysis NCA NeighborhoodComponentsAnalysis is a distance metric learning algorithm which aims to improve the accuracy of nearest neighbors classification compared to the standard Euclidean distance The algorithm directly maximizes a stochastic variant of the leaveoneout knearest neighbors KNN score on the training set It can also learn a lowdimensional linear projection of data that can be used for data visualization and fast classification

In the above illustrating figure we consider some points from a randomly generated dataset We focus on the stochastic KNN classification of point no 3 The thickness of a link between sample 3 and another point is proportional to their distance and can be seen as the relative weight or probability that a stochastic nearest neighbor prediction rule would assign to this point In the original space sample 3 has many stochastic neighbors from various classes so the right class is not very likely However in the projected space learned by NCA the only stochastic neighbors with non negligible weight are from the same class as sample 3 guaranteeing that the latter will be well classified See the mathematical formulation for more details

Classification

Combined with a nearest neighbors classifier KNeighborsClassifier NCA is attractive for classification because it can naturally handle multiclass problems without any increase in the model size and does not introduce additional parameters that require finetuning by the user

NCA classification has been shown to work well in practice for data sets of varying size and difficulty In contrast to related methods such as Linear Discriminant Analysis NCA does not make any assumptions about the class distributions The nearest neighbor classification can naturally produce highly irregular decision boundaries

To use this model for classification one needs to combine a NeighborhoodComponentsAnalysis instance that learns the optimal transformation with a KNeighborsClassifier instance that performs the classification in the projected space Here is an example using the two classes

```
from sklearn.neighbors import NeighborhoodComponentsAnalysis
KNeighborsClassifier
from sklearn.datasets import loadiris
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
X, y = loadiris(return_Xy=True)
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    stratify=y, test_size=0.7, random_state=42)
nca = NeighborhoodComponentsAnalysis(random_state=42)
knn = KNeighborsClassifier(n_neighbors=3)
ncapipeline = Pipeline([
    ('nca', nca),
    ('knn', knn)
])
ncapipeline.fit(X_train, y_train)
print(ncapipeline.score(X_test, y_test))
0.96190476
```

The plot shows decision boundaries for Nearest Neighbor Classification and Neighborhood Components Analysis classification on the iris dataset when training and scoring on only two features for visualisation purposes

Dimensionality reduction

NCA can be used to perform supervised dimensionality reduction The input data are projected onto a linear sub space consisting of the directions which minimize the NCA objective The desired dimensionality can be set using the parameter ncomponents For instance the following figure shows a comparison of dimensionality reduction with Principal Component Analysis sklearn.decomposition.PCA Linear Discriminant Analysis sklearn.discriminantanalysis.LinearDiscriminantAnalysis and Neighborhood Component Analysis NeighborhoodComponentsAnalysis on the Digits dataset a dataset with size 1797 and 64 The data set is split into a training and a test set of equal size then standardized For evaluation the 3nearest neighbor classification accuracy is computed on the 2dimensional projected points found by each method Each data sample belongs to one of 10 classes

Examples

- Comparing Nearest Neighbors with and without Neighborhood Components Analysis
- Dimensionality Reduction with Neighborhood Components Analysis
- Manifold learning on handwritten digits Locally Linear Embedding Isomap

Mathematical formulation

The goal of NCA is to learn an optimal linear transformation matrix of size ncomponents nfeatures which maximises the sum over all samples of the probability that is correctly classified ie

$$\arg \max_{W} -1 \sum_{i=0}^{n-1} \log p_i$$

with n samples and p_i the probability of sample i being correctly classified according to a stochastic nearest neighbors rule in the learned embedded space

$$p_i = \frac{\exp(-\min_{j \in C_i} \|x_i - x_j\|^2)}{\sum_{j \in C_i} \exp(-\|x_i - x_j\|^2)}$$

where C_i is the set of points in the same class as sample i and σ_j is the softmax over Euclidean distances in the embedded space

scikitlearn user guide Release 0213

Mahalanobis distance

NCA can be seen as learning a squared Mahalanobis distance metric

$$\|x - x'\|_M^2 = (x - x')^T M (x - x')$$

where M is a symmetric positive semidefinite matrix of size $n_{\text{features}} \times n_{\text{features}}$

Implementation

This implementation follows what is explained in the original paper¹ For the optimisation method it currently uses

scipy's LBFGSB with a full gradient computation at each iteration to avoid to tune the learning rate and provide

stable learning

See the examples below and the docstring of `NeighborhoodComponentsAnalysis` for further information

Complexity

Training

NCA stores a matrix of pairwise distances taking n_{samples}^2 memory Time complexity depends on the number of iterations done by the optimisation algorithm However one can set the maximum number of iterations with the argument `maxiter` For each iteration time complexity is $O(n_{\text{components}} \times n_{\text{samples}} \times \min(n_{\text{samples}}, n_{\text{features}}))$

$\times \min(n_{\text{samples}}, n_{\text{features}})$

Transform

Here the transform operation returns $n_{\text{components}}$ therefore its time complexity equals $n_{\text{components}}$

$n_{\text{features}} \times n_{\text{sample_test}}$ There is no added space complexity in the operation

References

317 Gaussian Processes

Gaussian Processes GP are a generic supervised learning method designed to solve regression and probabilistic classification problems

The advantages of Gaussian processes are

- The prediction interpolates the observations at least for regular kernels
- The prediction is probabilistic Gaussian so that one can compute empirical confidence intervals and decide based on those if one should refit online fitting adaptive fitting the prediction in some region of interest
- Versatile different kernels can be specified Common kernels are provided but it is also possible to specify custom kernels

¹"Neighbourhood Components Analysis" Advances in Neural Information Processing Systems V ol 17 May 2005 pp 513-520

256 Chapter 3 User Guide

The disadvantages of Gaussian processes include

- They are not sparse ie they use the whole samplesfeatures information to perform the prediction
- They lose efficiency in high dimensional spaces – namely when the number of features exceeds a few dozens

Gaussian Process Regression GPR

TheGaussianProcessRegressor implements Gaussian processes GP for regression purposes For this the

prior of the GP needs to be specified The prior mean is assumed to be constant and zero for normalizeyFalse or the training data’s mean for normalizeyTrue The prior’s covariance is specified by passing a kernel object

The hyperparameters of the kernel are optimized during fitting of GaussianProcessRegressor by maximizing the log

marginallikelihood LML based on the passed optimizer As the LML may have multiple local optima the

optimizer can be started repeatedly by specifying nrestartsoptimizer The first run is always conducted

starting from the initial hyperparameter values of the kernel subsequent runs are conducted from hyperparameter values that have been chosen randomly from the range of allowed values If the initial hyperparameters should be kept

fixedNone can be passed as optimizer

The noise level in the targets can be specified by passing it via the parameter alpha either globally as a scalar or

per datapoint Note that a moderate noise level can also be helpful for dealing with numeric issues during fitting as

it is effectively implemented as Tikhonov regularization ie by adding it to the diagonal of the kernel matrix An

alternative to specifying the noise level explicitly is to include a WhiteKernel component into the kernel which can

estimate the global noise level from the data see example below

The implementation is based on Algorithm 21 of RW2006 In addition to the API (Application Programming Interface) of standard scikitlearn estimators

GaussianProcessRegressor

- allows prediction without prior fitting based on the GP prior
- provides an additional method sampleyX which evaluates samples drawn from the GPR prior or posterior at given inputs
- exposes a method logmarginallikelihoodtheta which can be used externally for other ways of selecting hyperparameters eg via Markov chain Monte Carlo

GPR examples

GPR with noiselevel estimation

This example illustrates that GPR with a sumkernel including a WhiteKernel can estimate the noise level of data An

illustration of the logmarginallikelihood LML landscape shows that there exist two local maxima of LML

The first corresponds to a model with a high noise level and a large length scale which explains all variations in the data by noise

The second one has a smaller noise level and shorter length scale which explains most of the variation by the noise free functional relationship The second model has a higher likelihood however depending on the initial value for the hyperparameters the gradientbased optimization might also converge to the highnoise solution It is thus important

to repeat the optimization several times for different initializations

Comparison of GPR and Kernel Ridge Regression

Both kernel ridge regression KRR and GPR learn a target function by employing internally the “kernel trick” KRR

learns a linear function in the space induced by the respective kernel which corresponds to a nonlinear function in

the original space The linear function in the kernel space is chosen based on the meansquared error loss with ridge regularization GPR uses the kernel to define the covariance of a prior distribution over the target functions and uses

scikitlearn user guide Release 0213

the observed training data to define a likelihood function Based on Bayes theorem a Gaussian posterior distribution over target functions is defined whose mean is used for prediction

A major difference is that GPR can choose the kernel's hyperparameters based on gradient ascent on the marginal likelihood function while KRR needs to perform a grid search on a crossvalidated loss function meansquared error loss A further difference is that GPR learns a generative probabilistic model of the target function and can thus provide meaningful confidence intervals and posterior samples along with the predictions while KRR only provides predictions

The following figure illustrates both methods on an artificial dataset which consists of a sinusoidal target function and strong noise The figure compares the learned model of KRR and GPR based on a ExpSineSquared kernel which is suited for learning periodic functions The kernel's hyperparameters control the smoothness lengthscale and periodicity of the kernel periodicity Moreover the noise level of the data is learned explicitly by GPR by an additional WhiteKernel component in the kernel and by the regularization parameter alpha of KRR

The figure shows that both methods learn reasonable models of the target function GPR correctly identifies the periodicity of the function to be roughly $2\pi \times 628$ while KRR chooses the doubled periodicity 4π Besides that GPR provides reasonable confidence bounds on the prediction which are not available for KRR A major difference between the two methods is the time required for fitting and predicting while fitting KRR is fast in principle the gridsearch for hyperparameter optimization scales exponentially with the number of hyperparameters "curse of dimensionality" The gradientbased optimization of the parameters in GPR does not suffer from this exponential scaling and is thus considerable faster on this example with 3dimensional hyperparameter space The time for predicting is similar however generating the variance of the predictive distribution of GPR takes considerable longer than just predicting the mean

GPR on Mauna Loa CO2 data

This example is based on Section 543 of RW2006 It illustrates an example of complex kernel engineering and hyperparameter optimization using gradient ascent on the logmarginallikelihood The data consists of the monthly average atmospheric CO2 concentrations in parts per million by volume ppmv collected at the Mauna Loa Observatory in Hawaii between 1958 and 1997 The objective is to model the CO2 concentration as a function of the time t

The kernel is composed of several terms that are responsible for explaining different properties of the signal

scikitlearn user guide Release 0213

- a long term smooth rising trend is to be explained by an RBF (Radial Basis Function) kernel The RBF (Radial Basis Function) kernel with a large lengthscale enforces this component to be smooth it is not enforced that the trend is rising which leaves this choice to the GP The specific lengthscale and the amplitude are free hyperparameters
 - a seasonal component which is to be explained by the periodic ExpSineSquared kernel with a fixed periodicity of 1 year The lengthscale of this periodic component controlling its smoothness is a free parameter In order to allow decaying away from exact periodicity the product with an RBF (Radial Basis Function) kernel is taken The lengthscale of this RBF (Radial Basis Function) component controls the decay time and is a further free parameter
 - smaller medium term irregularities are to be explained by a RationalQuadratic kernel component whose lengthscale and alpha parameter which determines the diffuseness of the lengthscales are to be determined According to RW2006 these irregularities can better be explained by a RationalQuadratic than an RBF (Radial Basis Function) kernel component probably because it can accommodate several lengthscales
 - a “noise” term consisting of an RBF (Radial Basis Function) kernel contribution which shall explain the correlated noise components such as local weather phenomena and a WhiteKernel contribution for the white noise The relative amplitudes and the RBF (Radial Basis Function)’s length scale are further free parameters
- Maximizing the logmarginallikelihood after subtracting the target’s mean yields the following kernel with an LML of 83214

```
3442RBFlengthscale418
 3272RBFlengthscale180 ExpSineSquaredlengthscale144
periodicity1
 04462RationalQuadraticalpha177 lengthscale0957
 01972RBFlengthscale0138 WhiteKernelnoiselevel00336
```

Thus most of the target signal 344ppm is explained by a longterm rising trend lengthscale 418 years The periodic component has an amplitude of 327ppm a decay time of 180 years and a lengthscale of 144 The long decay time indicates that we have a locally very close to periodic seasonal component The correlated noise has an amplitude of 0197ppm with a length scale of 0138 years and a whitenoise contribution of 0197ppm Thus the overall noise level is very small indicating that the data can be very well explained by the model The figure shows also that the model makes very confident predictions until around 2015

Gaussian Process Classification GPC

TheGaussianProcessClassifier implements Gaussian processes GP for classification purposes more specifically for probabilistic classification where test predictions take the form of class probabilities GaussianProcessClassifier places a GP prior on a latent function f which is then squashed through a link function to obtain the probabilistic classification The latent function f is a so called nuisance function whose values are not observed and are not relevant by themselves Its purpose is to allow a convenient formulation of the model and f is removed integrated out during prediction GaussianProcessClassifier implements the logistic link function for which the integral cannot be computed analytically but is easily approximated in the binary case

In contrast to the regression setting the posterior of the latent function f is not Gaussian even for a GP prior since a Gaussian likelihood is inappropriate for discrete class labels Rather a nonGaussian likelihood corresponding to the logistic link function logit is used GaussianProcessClassifier approximates the nonGaussian posterior with a Gaussian based on the Laplace approximation More details can be found in Chapter 3 of RW2006

The GP prior mean is assumed to be zero The prior’s covariance is specified by passing a kernel object The hyperparameters of the kernel are optimized during fitting of GaussianProcessRegressor by maximizing the logmarginal likelihood LML based on the passed optimizer As the LML may have multiple local optima the optimizer can be started repeatedly by specifying nrestartsoptimizer The first run is always conducted starting from the initial hyperparameter values of the kernel subsequent runs are conducted from hyperparameter values that have been chosen randomly from the range of allowed values If the initial hyperparameters should be kept fixed None can be passed as optimizer

GaussianProcessClassifier supports multiclass classification by performing either oneversusrest or oneversusone based training and prediction In oneversusrest one binary Gaussian process classifier is fitted for each class which is trained to separate this class from the rest In “onevsone” one binary Gaussian process classifier is fitted for each pair of classes which is trained to separate these two classes The predictions of these binary predictors are combined into multiclass predictions See the section on multiclass classification for more details In the case of Gaussian process classification “onevsone” might be computationally cheaper since it has to solve many problems involving only a subset of the whole training set rather than fewer problems on the whole dataset Since Gaussian process classification scales cubically with the size of the dataset this might be considerably faster How ever note that “onevsone” does not support predicting probability estimates but only plain predictions Moreover note thatGaussianProcessClassifier does not yet implement a true multiclass Laplace approximation internally but as discussed above is based on solving several binary classification tasks internally which are combined using oneversusrest or oneversusone

GPC examples

Probabilistic predictions with GPC

This example illustrates the predicted probability of GPC for an RBF (Radial Basis Function) kernel with different choices of the hyperparameters The first figure shows the predicted probability of GPC with arbitrarily chosen hyperparameters and with the hyperparameters corresponding to the maximum logmarginallikelihood LML

While the hyperparameters chosen by optimizing LML have a considerable larger LML they perform slightly worse according to the logloss on test data The figure shows that this is because they exhibit a steep change of the class probabilities at the class boundaries which is good but have predicted probabilities close to 05 far away from the class boundaries which is bad This undesirable effect is caused by the Laplace approximation used internally by GPC

The second figure shows the logmarginallikelihood for different choices of the kernel’s hyperparameters highlighting the two choices of the hyperparameters used in the first figure by black dots

Illustration of GPC on the XOR dataset

This example illustrates GPC on XOR data Compared are a stationary isotropic kernel RBF (Radial Basis Function) and a nonstationary kernel DotProduct On this particular dataset the DotProduct kernel obtains considerably better results because the classboundaries are linear and coincide with the coordinate axes In practice however stationary kernels such asRBF often obtain better results

Gaussian process classification GPC on iris dataset

This example illustrates the predicted probability of GPC for an isotropic and anisotropic RBF (Radial Basis Function) kernel on a two dimensional version for the irisdataset This illustrates the applicability of GPC to nonbinary classification The anisotropic RBF (Radial Basis Function) kernel obtains slightly higher logmarginallikelihood by assigning different lengthscales to the two feature dimensions

Kernels for Gaussian Processes

Kernels also called “covariance functions” in the context of GPs are a crucial ingredient of GPs which determine the shape of prior and posterior of the GP They encode the assumptions on the function being learned by defining the “similarity” of two datapoints combined with the assumption that similar datapoints should have similar target values Two categories of kernels can be distinguished stationary kernels depend only on the distance of two datapoints and not on their absolute values $k(x, y) = k(x - y)$ and are thus invariant to translations in the input space

while nonstationary kernels depend also on the specific values of the datapoints Stationary kernels can further be subdivided into isotropic and anisotropic kernels where isotropic kernels are also invariant to rotations in the input space For more details we refer to Chapter 4 of RW2006

Gaussian Process Kernel API (Application Programming Interface)

The main usage of a Kernel is to compute the GP's covariance between datapoints For this the method call of the kernel can be called This method can either be used to compute the "autocovariance" of all pairs of datapoints in a 2d array X or the "crosscovariance" of all combinations of datapoints of a 2d array X with datapoints in a 2d array Y The following identity holds true for all kernels k except for the WhiteKernel $k_X(X, X) = Y_X Y_X$

If only the diagonal of the autocovariance is being used the method diag of a kernel can be called which is more computationally efficient than the equivalent call to call npdiagkX X kdiagX

Kernels are parameterized by a vector θ of hyperparameters These hyperparameters can for instance control length scales or periodicity of a kernel see below All kernels support computing analytic gradients of the kernel's autocovariance with respect to θ via setting evalgradient True in the call method This gradient is used by the Gaussian process both regressor and classifier in computing the gradient of the logmarginallikelihood which in turn is used to determine the value of θ which maximizes the logmarginallikelihood via gradient ascent For each hyperparameter the initial value and the bounds need to be specified when creating an instance of the kernel The current value of θ can be get and set via the property theta of the kernel object Moreover the bounds of the hyperparameters can be accessed by the property bounds of the kernel Note that both properties theta and bounds return logtransformed values of the internally used values since those are typically more amenable to gradientbased optimization The specification of each hyperparameter is stored in the form of an instance of Hyperparameter in the respective kernel Note that a kernel using a hyperparameter with name "x" must have the attributes selfx and selfxbounds

The abstract base class for all kernels is Kernel Kernel implements a similar interface as Estimator providing the methods getparams setparams and clone This allows setting kernel values also via meta estimators such as Pipeline or GridSearch Note that due to the nested structure of kernels by applying kernel operators see below the names of kernel parameters might become relatively complicated In general for a binary kernel operator parameters of the left operand are prefixed with k1 and parameters of the right operand with k2 An additional convenience method is clonewiththetatheta which returns a cloned version of the kernel

31 Supervised learning 267

scikitlearn user guide Release 0213

but with the hyperparameters set to theta An illustrative example
from sklearn.gaussianprocess.kernels import ConstantKernel RBF (Radial Basis Function)

```
kernel = ConstantKernel(constant_value=10, constant_value_bounds=(0, 100),  
    RBF_lengthscale=0.5, lengthscale_bounds=(0, 100), RBF_lengthscale=20,  
    lengthscale_bounds=(0, 100))  
for hyperparameter in kernel.hyperparameters:  
    print(hyperparameter)  
Hyperparameter name: k1.constant_value, value type: numeric, bounds: array([0,  
    100]), number of elements: 1, fixed: False  
Hyperparameter name: k1.lengthscale, value type: numeric, bounds: array([0,  
    100]), number of elements: 1, fixed: False  
Hyperparameter name: k2.lengthscale, value type: numeric, bounds: array([0, 100]),  
    number of elements: 1, fixed: False  
params = kernel.get_params()
```

```
for key in sorted(params):  
    print(ss, key, params[key])  
k1_12 RBF_lengthscale 0.5  
k1_1 1 2  
k1_1.constant_value 10  
k1_1.constant_value_bounds (0, 100)  
k1_2 RBF_lengthscale 0.5  
k1_2.lengthscale 0.5  
k1_2.lengthscale_bounds (0, 100)  
k2 RBF_lengthscale 2  
k2.lengthscale 20  
k2.lengthscale_bounds (0, 100)  
print(kernel.theta) Note: log transformed  
0.069314718 0.069314718  
print(kernel.bounds) Note: log transformed  
inf 230258509  
inf 230258509  
inf 230258509
```

All Gaussian process kernels are interoperable with sklearn.metrics.pairwise and vice versa instances of subclasses of Kernel can be passed as metric to pairwise.kernels from sklearn.metrics.pairwise Moreover kernel functions from pairwise can be used as GP kernels by using the wrapper class PairwiseKernel The only caveat is that the gradient of the hyperparameters is not analytic but numeric and all those kernels support only isotropic distances The parameter gamma is considered to be a hyperparameter and may be optimized The other kernel parameters are set directly at initialization and are kept fixed

Basic kernels

The ConstantKernel kernel can be used as part of a Product kernel where it scales the magnitude of the other factor kernel or as part of a Sum kernel where it modifies the mean of the Gaussian process It depends on a parameter γ It is defined as

$$k(x, y) = \gamma \exp(-\gamma \sqrt{\sum_{i=1}^n (x_i - y_i)^2})$$

The main usecase of the WhiteKernel kernel is as part of a sum kernel where it explains the noise component of the signal Tuning its parameter ν corresponds to estimating the noise level It is defined as

$$k(x, y) = \nu \text{ if } x = y \text{ else } 0$$

Kernel operators

Kernel operators take one or two base kernels and combine them into a new kernel The Sum kernel takes two kernels k_1 and k_2 and combines them via $k(x, y) = k_1(x, y) + k_2(x, y)$ The Product kernel takes two kernels k_1

and σ^2 and combines them via $\frac{1}{2\sigma^2} \exp(-\frac{1}{2\sigma^2} \|x - x'\|^2)$. The Exponentiation kernel takes one base kernel and a scalar parameter σ^2 and combines them via $\exp(-\sigma^2 \text{base_kernel}(x, x'))$.

Radialbasis function RBF (Radial Basis Function) kernel

The RBF kernel is a stationary kernel. It is also known as the “squared exponential” kernel. It is parameterized by a lengthscale parameter $\ell > 0$ which can either be a scalar isotropic variant of the kernel or a vector with the same number of dimensions as the inputs x . Anisotropic variant of the kernel. The kernel is given by

$$\frac{1}{(2\pi\ell^2)^{D/2}} \exp\left(-\frac{1}{2\ell^2} \|x - x'\|^2\right)$$

This kernel is infinitely differentiable which implies that GPs with this kernel as covariance function have mean square derivatives of all orders and are thus very smooth. The prior and posterior of a GP resulting from an RBF (Radial Basis Function) kernel are shown in the following figure.

Matérn kernel

The Matérn kernel is a stationary kernel and a generalization of the RBF (Radial Basis Function) kernel. It has an additional parameter ν which controls the smoothness of the resulting function. It is parameterized by a lengthscale parameter $\ell > 0$ which can either be a scalar isotropic variant of the kernel or a vector with the same number of dimensions as the inputs x . Anisotropic variant of the kernel. The kernel is given by

$$\frac{2^{D/2 + \nu} \Gamma(\nu)}{\sqrt{\ell} \Gamma(\nu + 1/2)} \left(\frac{\sqrt{2\nu}}{\ell}\right)^{2\nu} K_{\nu}\left(\frac{\sqrt{2\nu}}{\ell} \|x - x'\|\right)$$

As $\nu \rightarrow \infty$ the Matérn kernel converges to the RBF (Radial Basis Function) kernel. When $\nu = 1/2$ the Matérn kernel becomes identical to the absolute exponential kernel, i.e.

$$\frac{1}{2\ell} \exp\left(-\frac{\|x - x'\|}{\ell}\right)$$

In particular, $\nu = 3/2$

$$\frac{3}{4\ell^2} \exp\left(-\frac{\sqrt{3}}{\ell} \|x - x'\|\right)$$
$$\text{and } \nu = 5/2$$
$$\frac{15}{32\ell^5} \exp\left(-\frac{\sqrt{5}}{\ell} \|x - x'\|\right)$$

are popular choices for learning functions that are not infinitely differentiable as assumed by the RBF (Radial Basis Function) kernel but at least once $\nu = 3/2$ or twice differentiable $\nu = 5/2$.

The flexibility of controlling the smoothness of the learned function via ν allows adapting to the properties of the true underlying functional relation. The prior and posterior of a GP resulting from a Matérn kernel are shown in the following figure.

See RW2006, pp. 84 for further details regarding the different variants of the Matérn kernel.

Rational quadratic kernel

TheRationalQuadratic kernel can be seen as a scale mixture an infinite sum of RBF (Radial Basis Function) kernels with different characteristic lengthscales It is parameterized by a lengthscale parameter ℓ 0and a scale mixture parameter κ 0 Only the isotropic variant where ℓ is a scalar is supported at the moment The kernel is given by

$$k(x, y) = \frac{1}{2\sqrt{\kappa}} \int_0^\infty \exp\left(-\frac{\sqrt{\kappa} \|x - y\|}{2\ell} - \frac{t}{2\kappa}\right) dt$$

The prior and posterior of a GP resulting from a RationalQuadratic kernel are shown in the following figure

ExpSineSquared kernel

TheExpSineSquared kernel allows modeling periodic functions It is parameterized by a lengthscale parameter $\ell > 0$ and a periodicity parameter $p > 0$ Only the isotropic variant where ℓ is a scalar is supported at the moment

The kernel is given by

$$k(x, y) = \exp\left(-2 \sin^2\left(\frac{\pi}{p} \frac{\|x - y\|}{\ell}\right)\right)$$

The prior and posterior of a GP resulting from an ExpSineSquared kernel are shown in the following figure

DotProduct kernel

TheDotProduct kernel is nonstationary and can be obtained from linear regression by putting $\gamma_0=1$ priors on the coefficients of $\gamma_0=1$ and a prior of $\gamma_0=1$

on the bias The DotProduct kernel is invariant to a

rotation of the coordinates about the origin but not translations It is parameterized by a parameter γ_0

For $\gamma_0=1$

0 0

the kernel is called the homogeneous linear kernel otherwise it is inhomogeneous The kernel is given by

$$k(x, y) = \gamma_0 + \gamma_1 x \cdot y$$

TheDotProduct kernel is commonly combined with exponentiation An example with exponent 2 is shown in the following figure

scikitlearn user guide Release 0213

References

318 Cross decomposition

The cross decomposition module contains two main families of algorithms the partial least squares PLS and the canonical correlation analysis CCA

These families of algorithms are useful to find linear relations between two multivariate datasets the X and Y arguments of the fit method are 2D arrays

Cross decomposition algorithms find the fundamental relations between two matrices X and Y They are latent variable approaches to modeling the covariance structures in these two spaces They will try to find the multidimensional direction in the X space that explains the maximum multidimensional variance direction in the Y space PLS regression is particularly suited when the matrix of predictors has more variables than observations and when there is multicollinearity among X values By contrast standard regression will fail in these cases

Classes included in this module are PLSRegression PLSCanonical CCA and PLSSVD

Reference

- JA Wegelin A survey of Partial Least Squares PLS methods with emphasis on the two block case

Examples

31 Supervised learning 275

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes’ theorem with the “naive” assumption of conditional independence between every pair of features given the value of the class variable Bayes’ theorem states the following relationship given class variable c and dependent feature vector x_1 through x_n

$$P(c) \prod_{i=1}^n P(x_i | c)$$

Using the naive conditional independence assumption that

$$P(x_1, \dots, x_n | c) = \prod_{i=1}^n P(x_i | c)$$

for all c this relationship is simplified to

$$P(c) \prod_{i=1}^n P(x_i | c)$$

Since $P(c)$ is constant given the input we can use the following classification rule

$$P(c) \propto \prod_{i=1}^n P(x_i | c)$$

$$\downarrow$$
$$\arg \max_c \prod_{i=1}^n P(x_i | c)$$

and we can use Maximum A Posteriori MAP estimation to estimate $P(c)$ and $P(x_i | c)$ the former is then the relative frequency of class c in the training set

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of x_i

In spite of their apparently oversimplified assumptions naive Bayes classifiers have worked quite well in many real world situations famously document classification and spam filtering They require a small amount of training data to estimate the necessary parameters For theoretical reasons why naive Bayes works well and on which types of data it does see the references below

Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods The decoupling of the class conditional feature distributions means that each distribution can be independently estimated as a one dimensional distribution This in turn helps to alleviate problems stemming from the curse of dimensionality On the flip side although naive Bayes is known as a decent classifier it is known to be a bad estimator so the probability outputs from predict_proba are not to be taken too seriously

References

- H Zhang 2004 The optimality of Naive Bayes Proc FLAIRS

276 Chapter 3 User Guide

Gaussian Naive Bayes

GaussianNB implements the Gaussian Naive Bayes algorithm for classification The likelihood of the features is assumed to be Gaussian

$$\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

The parameters μ and σ are estimated using maximum likelihood

```
from sklearn import datasets
iris = datasets.load_iris()
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
ypred = gnb.fit(iris.data, iris.target).predict(iris.data)
print("Number of mislabeled points out of a total %d points: %d" %
      (iris.data.shape[0], iris.target != ypred).sum())
# Number of mislabeled points out of a total 150 points: 6
```

Multinomial Naive Bayes

MultinomialNB implements the naive Bayes algorithm for multinomially distributed data and is one of the two classic naive Bayes variants used in text classification where the data are typically represented as word vector counts although tfidf vectors are also known to work well in practice The distribution is parametrized by vectors θ and ϕ for each class c where ϕ is the number of features in text classification the size of the vocabulary and θ_c is the probability of feature i appearing in a sample belonging to class c

The parameters θ is estimated by a smoothed version of maximum likelihood ie relative frequency counting

$$\theta_{ci} = \frac{x_{ci} + 1}{\sum_j x_{cj} + V}$$

where x_{ci} is the number of times feature i appears in a sample of class c in the training set V and $V = \sum_i x_{ci}$ is the total count of all features for class c

The smoothing priors $\theta_{ci} \geq 0$ accounts for features not present in the learning samples and prevents zero probabilities in further computations Setting $\alpha = 1$ is called Laplace smoothing while $\alpha = 1/V$ is called Lidstone smoothing

Complement Naive Bayes

ComplementNB implements the complement naive Bayes CNB algorithm CNB is an adaptation of the standard multinomial naive Bayes MNB algorithm that is particularly suited for imbalanced data sets Specifically CNB uses statistics from the complement of each class to compute the model's weights The inventors of CNB show empirically that the parameter estimates for CNB are more stable than those for MNB Further CNB regularly outperforms MNB often by a considerable margin on text classification tasks The procedure for calculating the weights is as follows

$$w_c = \frac{\sum_{i \notin c} x_{ci}}{\sum_{i \notin c} x_{ci} + \alpha}$$
$$w_c = \log \frac{\sum_{i \notin c} x_{ci}}{\sum_{i \notin c} x_{ci} + \alpha}$$

where the summations are over all documents i not in class c α is either the count or tfidf value of term i in document c α is a smoothing hyperparameter like that found in MNB and $V = \sum_i x_{ci}$

The second normalization

scikitlearn user guide Release 0213

addresses the tendency for longer documents to dominate parameter estimates in MNB The classification rule is

$$\hat{c} = \arg \min_c \sum_{i=1}^n \mathbb{I}(y_i \neq c)$$

ie a document is assigned to the class that is the poorest complement match

References

- Rennie J D Shih L Teevan J Karger D R 2003 Tackling the poor assumptions of naive bayes text classifiers In ICML V ol 3 pp 616623

Bernoulli Naive Bayes

BernoulliNB implements the naive Bayes training and classification algorithms for data that is distributed according to multivariate Bernoulli distributions ie there may be multiple features but each one is assumed to be a binaryvalued Bernoulli boolean variable Therefore this class requires samples to be represented as binaryvalued feature vectors if handed any other kind of data a BernoulliNB instance may binarize its input depending on the binarize parameter

The decision rule for Bernoulli naive Bayes is based on

$$\hat{c} = \arg \min_c \sum_{i=1}^n \mathbb{I}(y_i \neq c)$$

which differs from multinomial NB’s rule in that it explicitly penalizes the nonoccurrence of a feature x_{ij} that is an indicator for class c where the multinomial variant would simply ignore a nonoccurring feature

In the case of text classification word occurrence vectors rather than word count vectors may be used to train and use this classifier BernoulliNB might perform better on some datasets especially those with shorter documents

It is advisable to evaluate both models if time permits

References

- CD Manning P Raghavan and H Schütze 2008 Introduction to Information Retrieval Cambridge University Press pp 234265
- A McCallum and K Nigam 1998 A comparison of event models for Naive Bayes text classification Proc AAAI/ICML98 Workshop on Learning for Text Categorization pp 4148
- V Metsis I Androutsopoulos and G Paliouras 2006 Spam filtering with Naive Bayes – Which Naive Bayes 3rd Conf on Email and AntiSpam CEAS

Outofcore naive Bayes model fitting

Naive Bayes models can be used to tackle large scale classification problems for which the full training set might not fit in memory To handle this case MultinomialNB BernoulliNB andGaussianNB expose apartialfit method that can be used incrementally as done with other classifiers as demonstrated in Outofcore classification of text documents All naive Bayes classifiers support sample weighting

Contrary to the fit method the first call to partialfit needs to be passed the list of all the expected class labels

For an overview of available strategies in scikitlearn see also the outofcore learning documentation

scikitlearn user guide Release 0213

Note Thepartialfit method call of naive Bayes models introduces some computational overhead It is recommended to use data chunk sizes that are as large as possible that is as the available RAM allows

3110 Decision Trees

Decision Trees DTs are a nonparametric supervised learning method used for classification andregression The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features

For instance in the example below decision trees learn from data to approximate a sine curve with a set of ifthenelse decision rules The deeper the tree the more complex the decision rules and the fitter the model

Some advantages of decision trees are

- Simple to understand and to interpret Trees can be visualised
- Requires little data preparation Other techniques often require data normalisation dummy variables need to be created and blank values to be removed Note however that this module does not support missing values
- The cost of using the tree ie predicting data is logarithmic in the number of data points used to train the tree
- Able to handle both numerical and categorical data Other techniques are usually specialised in analysing datasets that have only one type of variable See algorithms for more information
- Able to handle multioutput problems
- Uses a white box model If a given situation is observable in a model the explanation for the condition is easily explained by boolean logic By contrast in a black box model eg in an artificial neural network results may be more difficult to interpret
- Possible to validate a model using statistical tests That makes it possible to account for the reliability of the model

31 Supervised learning 279

scikitlearn user guide Release 0213

- Performs well even if its assumptions are somewhat violated by the true model from which the data were generated

The disadvantages of decision trees include

- Decisiontree learners can create overcomplex trees that do not generalise the data well This is called overfitting Mechanisms such as pruning not currently supported setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem
- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated This problem is mitigated by using decision trees within an ensemble
- The problem of learning an optimal decision tree is known to be NPcomplete under several aspects of optimality and even for simple concepts Consequently practical decisiontree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node Such algorithms cannot guarantee to return the globally optimal decision tree This can be mitigated by training multiple trees in an ensemble learner where the features and samples are randomly sampled with replacement
- There are concepts that are hard to learn because decision trees do not express them easily such as XOR parity or multiplexer problems
- Decision tree learners create biased trees if some classes dominate It is therefore recommended to balance the dataset prior to fitting with the decision tree

Classification

DecisionTreeClassifier is a class capable of performing multiclass classification on a dataset As with other classifiers DecisionTreeClassifier takes as input two arrays an array X sparse or dense of sizensamples nfeatures holding the training samples and an array Y of integer values size nsamples holding the class labels for the training samples

```
from sklearn import tree
X = [[0, 0, 1, 1],
     [0, 1, 0, 1]]
Y = [0, 1]
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)
```

After being fitted the model can then be used to predict the class of samples

```
clf.predict([2, 2])
array([1])
Alternatively the probability of each class can be predicted which is the fraction of training samples of the same class in a leaf
clf.predict_proba([2, 2])
array([[0, 1]])
```

DecisionTreeClassifier is capable of both binary where the labels are 1 1 classification and multiclass where the labels are 0 K1 classification

Using the Iris dataset we can construct a tree as follows

```
from sklearn.datasets import load_iris
from sklearn import tree
iris = load_iris()
clf = tree.DecisionTreeClassifier()
clf = clf.fit(iris.data, iris.target)
```

scikitlearn user guide Release 0213

Once trained you can plot the tree with the `plottree` function

`treeplottree(clf.fit(iris.data, iris.target))`

We can also export the tree in Graphviz format using the `exportgraphviz` exporter. If you use the conda package manager, the graphviz binaries

and the python package can be installed with

`conda install python-graphviz`

Alternatively, binaries for graphviz can be downloaded from the graphviz project homepage and the Python wrapper

installed from pypi with `pip install graphviz`

Below is an example graphviz export of the above tree trained on the entire iris dataset; the results are saved in an output file `iris.pdf`:

```
import graphviz
dotdata = tree.export_graphviz(clf, out_file=None)
graph = graphviz.Source(dotdata)
graph.render('iris')
```

The `export_graphviz` exporter also supports a variety of aesthetic options including coloring nodes by their class or value for regression and using explicit variable and class names if desired. Jupyter notebooks also render these plots inline automatically.

```
dotdata = tree.export_graphviz(clf, out_file=None,
                              feature_names=iris.feature_names,
                              class_names=iris.target_names,
                              filled=True, rounded=True,
                              special_characters=True)
graph = graphviz.Source(dotdata)
graph
```

31 Supervised learning 281

scikitlearn user guide Release 0213

Alternatively the tree can also be exported in textual format with the function `exporttext`. This method doesn't require the installation of external libraries and is more compact.

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree.export import export_text

iris = load_iris()
X = iris.data
y = iris.target

decisiontree = DecisionTreeClassifier(random_state=0, max_depth=2)
decisiontree.fit(X, y)

r = export_text(decisiontree, feature_names=iris.feature_names)
print(r)

petal width cm 0.80
class 0
petal width cm 0.80
petal width cm 1.75
class 1
petal width cm 1.75
class 2
```

Examples

- Plot the decision surface of a decision tree on the iris dataset
- Understanding the decision tree structure

Regression

Decision trees can also be applied to regression problems using the `DecisionTreeRegressor` class.

scikitlearn user guide Release 0213

As in the classification setting the fit method will take as argument arrays X and y only that in this case y is expected to have floating point values instead of integer values

```
from sklearn import tree
X = [[0, 0, 2, 2],
     [0.5, 2.5]]
clf = tree.DecisionTreeRegressor()
clf.fit(X, y)
clf.predict([[1, 1]])
array([0.5])
```

Examples

•Decision Tree Regression

Multioutput problems

A multioutput problem is a supervised learning problem with several outputs to predict that is when Y is a 2d array of size n_samples x n_outputs

When there is no correlation between the outputs a very simple way to solve this kind of problem is to build n independent models ie one for each output and then to use those models to independently predict each one of the n outputs However because it is likely that the output values related to the same input are themselves correlated an often better way is to build a single model capable of predicting simultaneously all n outputs First it requires lower training time since only a single estimator is built Second the generalization accuracy of the resulting estimator may often be increased

With regard to decision trees this strategy can readily be used to support multioutput problems This requires the following changes

- Store n output values in leaves instead of 1
- Use splitting criteria that compute the average reduction across all n outputs

This module offers support for multioutput problems by implementing this strategy in both

DecisionTreeClassifier and DecisionTreeRegressor If a decision tree is fit on an output array Y of size n_samples x n_outputs then the resulting estimator will

- Output n_output values upon predict
- Output a list of n_output arrays of class probabilities upon predict_proba

The use of multioutput trees for regression is demonstrated in Multioutput Decision Tree Regression In this example the input X is a single real value and the outputs Y are the sine and cosine of X

The use of multioutput trees for classification is demonstrated in Face completion with a multioutput estimators In this example the inputs X are the pixels of the upper half of faces and the outputs Y are the pixels of the lower half of those faces

Examples

•Multioutput Decision Tree Regression

•Face completion with a multioutput estimators

References

- M Dumont et al Fast multiclass image annotation with random subwindows and multiple output randomized trees International Conference on Computer Vision Theory and Applications 2009

Complexity

In general the run time cost to construct a balanced binary tree is $O(n \log n)$ and query time $O(\log n)$. Although the tree construction algorithm attempts to generate balanced trees they will not always be balanced. Assuming that the subtrees remain approximately balanced the cost at each node consists of searching through n to find the feature that offers the largest reduction in entropy. This has a cost of $O(n \log n)$ at each node leading to a total cost over the entire trees by summing the cost at each node of $O(n^2 \log n)$.

Scikitlearn offers a more efficient implementation for the construction of decision trees. A naive implementation as above would recompute the class label histograms for classification or the means for regression at for each new split point along a given feature. Presorting the feature over all relevant samples and retaining a running label count will reduce the complexity at each node to $O(n \log n)$ which results in a total cost of $O(n^2 \log n)$. This is an option for all tree based algorithms. By default it is turned on for gradient boosting where in general it makes training faster but turned off for all other algorithms as it tends to slow down training when training deep trees.

Tips on practical use

- Decision trees tend to overfit on data with a large number of features. Getting the right ratio of samples to number of features is important since a tree with few samples in high dimensional space is very likely to overfit.

scikitlearn user guide Release 0213

- Consider performing dimensionality reduction PCA (Principal Component Analysis) ICA or Feature selection beforehand to give your tree a better chance of finding features that are discriminative
- Understanding the decision tree structure will help in gaining more insights about how the decision tree makes predictions which is important for understanding the important features in the data
- Visualise your tree as you are training by using the export function Use maxdepth3 as an initial tree depth to get a feel for how the tree is fitting to your data and then increase the depth
- Remember that the number of samples required to populate the tree doubles for each additional level the tree grows to Use maxdepth to control the size of the tree to prevent overfitting
- Use minsamplessplit or minsamplesleaf to ensure that multiple samples inform every decision in the tree by controlling which splits will be considered A very small number will usually mean the tree will overfit whereas a large number will prevent the tree from learning the data Try minsamplesleaf5 as an initial value If the sample size varies greatly a float number can be used as percentage in these two parameters While minsamplessplit can create arbitrarily small leaves minsamplesleaf guarantees that each leaf has a minimum size avoiding low variance overfit leaf nodes in regression problems For classification with few classes minsamplesleaf1 is often the best choice
- Balance your dataset before training to prevent the tree from being biased toward the classes that are dominant Class balancing can be done by sampling an equal number of samples from each class or preferably by normalizing the sum of the sample weights sampleweight for each class to the same value Also note that weightbased prepruning criteria such as minweightfractionleaf will then be less biased toward dominant classes than criteria that are not aware of the sample weights like minsamplesleaf
- If the samples are weighted it will be easier to optimize the tree structure using weightbased prepruning criterion such as minweightfractionleaf which ensure that leaf nodes contain at least a fraction of the overall sum of the sample weights
- All decision trees use npfloat32 arrays internally If training data is not in this format a copy of the dataset will be made
- If the input matrix X is very sparse it is recommended to convert to sparse cscmatrix before calling fit and sparsecsrmatrix before calling predict Training time can be orders of magnitude faster for a sparse matrix input compared to a dense matrix when features have zero values in most of the samples

Tree algorithms ID3 C45 C50 and CART

What are all the various decision tree algorithms and how do they differ from each other Which one is implemented in scikitlearn

ID3 Iterative Dichotomiser 3 was developed in 1986 by Ross Quinlan The algorithm creates a multiway tree finding for each node ie in a greedy manner the categorical feature that will yield the largest information gain for categorical targets Trees are grown to their maximum size and then a pruning step is usually applied to improve the ability of the tree to generalise to unseen data

C45 is the successor to ID3 and removed the restriction that features must be categorical by dynamically defining a discrete attribute based on numerical variables that partitions the continuous attribute value into a discrete set of intervals C45 converts the trained trees ie the output of the ID3 algorithm into sets of ifthen rules These accuracy of each rule is then evaluated to determine the order in which they should be applied Pruning is done by removing a rule's precondition if the accuracy of the rule improves without it

C50 is Quinlan's latest version release under a proprietary license It uses less memory and builds smaller rulesets than C45 while being more accurate

CART Classification and Regression Trees is very similar to C45 but it differs in that it supports numerical target variables regression and does not compute rule sets CART constructs binary trees using the feature and threshold that yield the largest information gain at each node

31 Supervised learning 287

scikitlearn uses an optimised version of the CART algorithm however scikitlearn implementation does not support categorical variables for now

Mathematical formulation

Given training vectors $x_i \in \mathbb{R}^d$ $i=1 \dots n$ and a label vector $y \in \mathbb{R}$ a decision tree recursively partitions the space such that the samples with the same labels are grouped together

Let the data at node j be represented by D_j For each candidate split $s = (f, \tau)$ consisting of a feature f and threshold τ partition the data into D_{jL} and D_{jR} subsets

$$D_{jL} = \{x_i \in D_j \mid f(x_i) \leq \tau\}$$
$$D_{jR} = \{x_i \in D_j \mid f(x_i) > \tau\}$$

The impurity at j is computed using an impurity function i the choice of which depends on the task being solved classification or regression

$$i(D_j) = i(D_{jL}, D_{jR})$$
$$i(D_j) = i(D_{jL}) + i(D_{jR})$$

Select the parameters that minimises the impurity

$s = \arg\min_{s \in S} i(D_j, s)$
Recurse for subsets D_{jL} and D_{jR} until the maximum allowable depth is reached \max_depth or $n \leq 1$

Classification criteria

If a target is a classification outcome taking on values $0, 1, \dots, K-1$ for node j representing a region R_j with n_j observations let

$$p_{jk} = \frac{1}{n_j} \sum_{i \in R_j} \mathbb{1}_{y_i = k}$$

be the proportion of class k observations in node j

Common measures of impurity are Gini

$$Gini(D_j) = \sum_{k=0}^{K-1} p_{jk}(1 - p_{jk})$$

Entropy

$$Entropy(D_j) = -\sum_{k=0}^{K-1} p_{jk} \log(p_{jk})$$

and Misclassification

$$Misclassification(D_j) = 1 - \max_k p_{jk}$$

where p_{jk} is the training data in node j

Regression criteria

If the target is a continuous value then for node n representing a region R_n with N_n observations common criteria to minimise as for determining locations for future splits are Mean Squared Error which minimizes the L2 error using mean values at terminal nodes and Mean Absolute Error which minimizes the L1 error using median values at terminal nodes

Mean Squared Error

$$\frac{1}{N_n} \sum_{i \in R_n} (y_i - \hat{y}_n)^2$$

Mean Absolute Error

$$\frac{1}{N_n} \sum_{i \in R_n} |y_i - \hat{y}_n|$$

where R_n is the training data in node n

References

- https://en.wikipedia.org/wiki/Decision_tree_learning
- https://en.wikipedia.org/wiki/Predictive_analytics
- L Breiman J Friedman R Olshen and C Stone Classification and Regression Trees Wadsworth Belmont CA 1984
- JR Quinlan C4 5 programs for machine learning Morgan Kaufmann 1993
- T Hastie R Tibshirani and J Friedman Elements of Statistical Learning Springer 2009

3111 Ensemble methods

The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability robustness over a single estimator

Two families of ensemble methods are usually distinguished

- In averaging methods the driving principle is to build several estimators independently and then to average their predictions On average the combined estimator is usually better than any of the single base estimator because its variance is reduced

Examples Bagging methods Forests of randomized trees

- By contrast in boosting methods base estimators are built sequentially and one tries to reduce the bias of the combined estimator The motivation is to combine several weak models to produce a powerful ensemble

Examples AdaBoost Gradient Tree Boosting

Bagging metaestimator

In ensemble algorithms bagging methods form a class of algorithms which build several instances of a blackbox estimator on random subsets of the original training set and then aggregate their individual predictions to form a final prediction These methods are used as a way to reduce the variance of a base estimator eg a decision tree by introducing randomization into its construction procedure and then making an ensemble out of it In many cases bagging methods constitute a very simple way to improve with respect to a single model without making it necessary to adapt the underlying base algorithm As they provide a way to reduce overfitting bagging methods work best with strong and complex models eg fully developed decision trees in contrast with boosting methods which usually work best with weak models eg shallow decision trees

Bagging methods come in many flavours but mostly differ from each other by the way they draw random subsets of the training set

- When random subsets of the dataset are drawn as random subsets of the samples then this algorithm is known as Pasting B1999
- When samples are drawn with replacement then the method is known as Bagging B1996
- When random subsets of the dataset are drawn as random subsets of the features then the method is known as Random Subspaces H1998
- Finally when base estimators are built on subsets of both samples and features then the method is known as Random Patches LG2012

In scikitlearn bagging methods are offered as a unified BaggingClassifier metaestimator resp BaggingRegressor taking as input a userspecified base estimator along with parameters specifying the strategy to draw random subsets In particular maxsamples andmaxfeatures control the size of the subsets in terms of samples and features while bootstrap andbootstrapfeatures control whether samples and features are drawn with or without replacement When using a subset of the available samples the generalization accuracy can be estimated with the outofbag samples by setting oobscoreTrue As an example the snippet below illustrates how to instantiate a bagging ensemble of KNeighborsClassifier base estimators each built on random subsets of 50 of the samples and 50 of the features

```
from sklearnensemble import BaggingClassifier
from sklearnneighbors import KNeighborsClassifier
bagging = BaggingClassifier(KNeighborsClassifier
    maxsamples05 maxfeatures05
```

Examples

- Single estimator versus bagging biasvariance decomposition

References

Forests of randomized trees

The sklearnensemble module includes two averaging algorithms based on randomized decision trees the RandomForest algorithm and the ExtraTrees method Both algorithms are perturbandcombine techniques B1998 specifically designed for trees This means a diverse set of classifiers is created by introducing randomness in the classifier construction The prediction of the ensemble is given as the averaged prediction of the individual classifiers

290 Chapter 3 User Guide

scikitlearn user guide Release 0213

As other classifiers forest classifiers have to be fitted with two arrays a sparse or dense array X of size nsamples nfeatures holding the training samples and an array Y of size nsamples holding the target values class labels for the training samples

```
from sklearnensemble import RandomForestClassifier
X 0 0 1 1
Y 0 1
clf RandomForestClassifiernestimators10
clf clffitX Y
```

Like decision trees forests of trees also extend to multioutput problems if Y is an array of size nsamples noutputs

Random Forests

In random forests see RandomForestClassifier andRandomForestRegressor classes each tree in the ensemble is built from a sample drawn with replacement ie a bootstrap sample from the training set Furthermore when splitting each node during the construction of a tree the best split is found either from all input features or a random subset of size maxfeatures See the parameter tuning guidelines for more details The purpose of these two sources of randomness is to decrease the variance of the forest estimator Indeed individual decision trees typically exhibit high variance and tend to overfit The injected randomness in forests yield decision trees with somewhat decoupled prediction errors By taking an average of those predictions some errors can cancel out Random forests achieve a reduced variance by combining diverse trees sometimes at the cost of a slight increase in bias In practice the variance reduction is often significant hence yielding an overall better model In contrast to the original publication B2001 the scikitlearn implementation combines classifiers by averaging their probabilistic prediction instead of letting each classifier vote for a single class

Extremely Randomized Trees

In extremely randomized trees see ExtraTreesClassifier andExtraTreesRegressor classes randomness goes one step further in the way splits are computed As in random forests a random subset of candidate features is used but instead of looking for the most discriminative thresholds thresholds are drawn at random for each candidate feature and the best of these randomlygenerated thresholds is picked as the splitting rule This usually allows to reduce the variance of the model a bit more at the expense of a slightly greater increase in bias

```
from sklearnmodelselection import crossvalscore
from sklearndatasets import makeblobs
from sklearnensemble import RandomForestClassifier
from sklearnensemble import ExtraTreesClassifier
from sklearnntree import DecisionTreeClassifier
X y makeblobsnsamples10000 nfeatures10 centers100
randomstate0
clf DecisionTreeClassifiermaxdepth None minsamplesplit2
randomstate0
scores crossvalscoreclf X y cv5
scoresmean
098
clf RandomForestClassifiernestimators10 maxdepth None
minsamplesplit2 randomstate0
scores crossvalscoreclf X y cv5
31 Supervised learning 291
```

```
scoresmean
0999
clf ExtraTreesClassifiernestimators10 maxdepth None
  minsamplesplit2 randomstate0
scores crossvalscoreclf X y cv5
scoresmean 0999
True
```

Parameters

The main parameters to adjust when using these methods is nestimators andmaxfeatures. The former is the number of trees in the forest. The larger the better but also the longer it will take to compute. In addition note that results will stop getting significantly better beyond a critical number of trees. The latter is the size of the random subsets of features to consider when splitting a node. The lower the greater the reduction of variance but also the greater the increase in bias. Empirical good default values are maxfeaturesNone always considering all features instead of a random subset for regression problems and maxfeaturesqrt using a random subset of sizesqrtnfeatures for classification tasks where nfeatures is the number of features in the data. Good results are often achieved when setting maxdepthNone in combination with minsamplesplit2 ie when fully developing the trees. Bear in mind though that these values are usually not optimal and might result in models that consume a lot of RAM. The best parameter values should always be crossvalidated. In addition note that in random forests bootstrap samples are used by default bootstrapTrue while the default strategy for extratrees is to use the whole dataset bootstrapFalse. When using bootstrap sampling the generalization accuracy can be estimated on the left out or outofbag samples. This can be enabled by setting oobscoreTrue. Note The size of the model with the default parameters is $O(n \log n)$ where n is the number of trees and m is the number of samples. In order to reduce the size of the model you can change these parameters minsamplesplit maxleafnodes maxdepth andminsamplesleaf.

292 Chapter 3 User Guide

Parallelization

Finally this module also features the parallel construction of the trees and the parallel computation of the predictions through the `njobs` parameter. If `njobs=k` then computations are partitioned into `k` jobs and run on `k` cores of the machine. If `njobs=1` then all cores available on the machine are used. Note that because of interprocess communication overhead the speedup might not be linear, ie using `k` jobs will unfortunately not be `k` times as fast. Significant speedup can still be achieved though when building a large number of trees or when building a single tree requires a fair amount of time eg on large datasets.

Examples

- Plot the decision surfaces of ensembles of trees on the iris dataset
- Pixel importances with a parallel forest of trees
- Face completion with a multioutput estimators

References

- P. Geurts, D. Ernst and L. Wehenkel “Extremely randomized trees” Machine Learning 63:1–342, 2006

Feature importance evaluation

The relative rank ie depth of a feature used as a decision node in a tree can be used to assess the relative importance of that feature with respect to the predictability of the target variable. Features used at the top of the tree contribute to the final prediction decision of a larger fraction of the input samples. The expected fraction of the samples they contribute to can thus be used as an estimate of the relative importance of the features. In scikitlearn the fraction of samples a feature contributes to is combined with the decrease in impurity from splitting them to create a normalized estimate of the predictive power of that feature.

By averaging the estimates of predictive ability over several randomized trees one can reduce the variance of such an estimate and use it for feature selection. This is known as the mean decrease in impurity or MDI. Refer to L2014 for more information on MDI and feature importance evaluation with Random Forests.

The following example shows a color-coded representation of the relative importances of each individual pixel for a face recognition task using an `ExtraTreesClassifier` model.

In practice those estimates are stored as an attribute named `feature_importances_` on the fitted model. This is an array with shape `n_features` whose values are positive and sum to 1.0. The higher the value the more important is the contribution of the matching feature to the prediction function.

Examples

- Pixel importances with a parallel forest of trees
- Feature importances with forests of trees

References

scikitlearn user guide Release 0213

Totally Random Trees Embedding

RandomTreesEmbedding implements an unsupervised transformation of the data Using a forest of completely random trees RandomTreesEmbedding encodes the data by the indices of the leaves a data point ends up in This index is then encoded in a oneofK manner leading to a high dimensional sparse binary coding This coding can be computed very efficiently and can then be used as a basis for other learning tasks The size and sparsity of the code can be influenced by choosing the number of trees and the maximum depth per tree For each tree in the ensemble the coding contains one entry of one The size of the coding is at most nestimators 2maxdepth the maximum number of leaves in the forest

As neighboring data points are more likely to lie within the same leaf of a tree the transformation performs an implicit nonparametric density estimation

Examples

- Hashing feature transformation using Totally Random Trees
- Manifold learning on handwritten digits Locally Linear Embedding Isomap compares nonlinear dimensionality reduction techniques on handwritten digits
- Feature transformations with ensembles of trees compares supervised and unsupervised tree based feature transformations

See also

Manifold learning techniques can also be useful to derive nonlinear representations of feature space also these approaches focus also on dimensionality reduction

AdaBoost

The module sklearnensemble includes the popular boosting algorithm AdaBoost introduced in 1995 by Freund and Schapire FS1995

The core principle of AdaBoost is to fit a sequence of weak learners ie models that are only slightly better than random guessing such as small decision trees on repeatedly modified versions of the data The predictions from all of them are then combined through a weighted majority vote or sum to produce the final prediction The data modifications at each socalled boosting iteration consist of applying weights w_t to each of the training samples Initially those weights are all set to $w_1 = 1$ so that the first step simply trains a weak learner on the original data For each successive iteration the sample weights are individually modified and the learning algorithm is reapplied to the reweighted data At a given step those training examples that were incorrectly predicted by the boosted model induced at the previous step have their weights increased whereas the weights are decreased for those that were predicted correctly As iterations proceed examples that are difficult to predict receive everincreasing influence Each subsequent weak learner is thereby forced to concentrate on the examples that are missed by the previous ones in the sequence HTF

AdaBoost can be used both for classification and regression problems

- For multiclass classification AdaBoostClassifier implements AdaBoostSAMME and AdaBoostSAMMER ZZRH2009

- For regression AdaBoostRegressor implements AdaBoostR2 D1997

Usage

The following example shows how to fit an AdaBoost classifier with 100 weak learners

```
from sklearnmodelselection import crossvalscore
from sklearndatasets import loadiris
from sklearnensemble import AdaBoostClassifier
```

scikitlearn user guide Release 0213

```
iris loadiris
clf AdaBoostClassifiernestimators100
scores crossvalscoreclf irisdata iristarget cv5
scoresmean
09
```

The number of weak learners is controlled by the parameter nestimators Thelearningrate parameter controls the contribution of the weak learners in the final combination By default weak learners are decision stumps Different weak learners can be specified through the baseestimator parameter The main parameters to tune to obtain good results are nestimators and the complexity of the base estimators eg its depth maxdepth or minimum required number of samples to consider a split minsamplesplit

Examples

- Discrete versus Real AdaBoost compares the classification error of a decision stump decision tree and a boosted decision stump using AdaBoostSAMME and AdaBoostSAMMER
- Multiclass AdaBoosted Decision Trees shows the performance of AdaBoostSAMME and AdaBoost SAMMER on a multiclass problem
- Twoclass AdaBoost shows the decision boundary and decision function values for a nonlinearly separable twoclass problem using AdaBoostSAMME
- Decision Tree Regression with AdaBoost demonstrates regression with the AdaBoostR2 algorithm

References

Gradient Tree Boosting

Gradient Tree Boosting or Gradient Boosted Regression Trees GBRT is a generalization of boosting to arbitrary differentiable loss functions GBRT is an accurate and effective offtheshelf procedure that can be used for both regression and classification problems Gradient Tree Boosting models are used in a variety of areas including Web search ranking and ecology

The advantages of GBRT are

- Natural handling of data of mixed type heterogeneous features
- Predictive power
- Robustness to outliers in output space via robust loss functions

The disadvantages of GBRT are

- Scalability due to the sequential nature of boosting it can hardly be parallelized

The module sklearnensemble provides methods for both classification and regression via gradient boosted regression trees

Note Scikitlearn 021 introduces two new experimental implementation of gradient boosting trees namely HistGradientBoostingClassifier andHistGradientBoostingRegressor inspired by Light

GBM These fast estimators first bin the input samples Xinto integervalue bins typically 256 bins which tremen dously reduces the number of splitting points to consider and allow the algorithm to leverage integerbased data structures histograms instead of relying on sorted continuous values

scikitlearn user guide Release 0213

The new histogrambased estimators can be orders of magnitude faster than their continuous counterparts when the number of samples is larger than tens of thousands of samples The API (Application Programming Interface) of these new estimators is slightly different and some of the features from GradientBoostingClassifier and GradientBoostingRegressor are not yet supported

These new estimators are still experimental for now their predictions and their API (Application Programming Interface) might change without any deprecation cycle To use them you need to explicitly import enablehistgradientboosting

```
explicitly require this experimental feature
from sklearn.experimental import enablehistgradientboosting
now you can import normally from ensemble
from sklearn.ensemble import HistGradientBoostingClassifier
```

The following guide focuses on GradientBoostingClassifier and GradientBoostingRegressor only which might be preferred for small sample sizes since binning may lead to split points that are too approximate in this setting

Classification

GradientBoostingClassifier supports both binary and multiclass classification The following example shows how to fit a gradient boosting classifier with 100 decision stumps as weak learners

```
from sklearn.datasets import make_hastie102
from sklearn.ensemble import GradientBoostingClassifier
X, y = make_hastie102(random_state=0)
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.33,
                                                    random_state=0)
clf = GradientBoostingClassifier(n_estimators=100, learning_rate=10,
                                max_depth=1, random_state=0)
clf.fit(X_train, y_train)
clf.score(X_test, y_test)
```

The number of weak learners ie regression trees is controlled by the parameter n_estimators The size of each tree can be controlled either by setting the tree depth via max_depth or by setting the number of leaf nodes via max_leaf_nodes The learning_rate is a hyperparameter in the range [0, 1] that controls overfitting via shrinkage

Note Classification with more than 2 classes requires the induction of n_classes regression trees at each iteration thus the total number of induced trees equals n_classes * n_estimators For datasets with a large number of classes we strongly recommend to use RandomForestClassifier as an alternative to GradientBoostingClassifier

Regression

GradientBoostingRegressor supports a number of different loss functions for regression which can be specified via the argument loss the default loss function for regression is least squares

```
import numpy as np
from sklearn.metrics import mean_squared_error
from sklearn.datasets import make_friedman1
```

scikitlearn user guide Release 0213

```
from sklearnensemble import GradientBoostingRegressor
X, y = makefriedman1(nsamples=1200, randomstate=0, noise=10)
Xtrain, Xtest, ytrain, ytest = X[0:1000], X[1000:1200], y[0:1000], y[1000:1200]
est = GradientBoostingRegressor(n_estimators=100, learning_rate=0.1,
                                max_depth=1, random_state=0, loss='ls', fit_params={'Xtrain': Xtrain, 'ytrain': ytrain})
est.fit(Xtrain, ytrain)
ytest_pred = est.predict(Xtest)
500
```

The figure below shows the results of applying GradientBoostingRegressor with least squares loss and 500 base learners to the Boston house price dataset. `sklearn.datasets.load_boston`. The plot on the left shows the train and test error at each iteration. The train error at each iteration is stored in the `train_score` attribute of the gradient boosting model. The test error at each iteration can be obtained via the `staged_predict` method which returns a generator that yields the predictions at each stage. Plots like these can be used to determine the optimal number of trees (i.e. `n_estimators`) by early stopping. The plot on the right shows the feature importances which can be obtained via the `feature_importances_` property.

Examples

- Gradient Boosting regression
- Gradient Boosting Out-of-Bag estimates

Fitting additional weak learners

Both `GradientBoostingRegressor` and `GradientBoostingClassifier` support

`warm_start=True` which allows you to add more estimators to an already fitted model.

estsetparamsnestimators200 warmstart True set warmstart and new
'→nr of trees

estfitXtrain ytrain fit additional 100 trees to est
meansquarederrorytest estpredictXtest
384

Controlling the tree size

The size of the regression tree base learners defines the level of variable interactions that can be captured by the gradient boosting model In general a tree of depth h can capture interactions of order h There are two ways in which the size of the individual regression trees can be controlled

If you specify maxdepthh then complete binary trees of depth h will be grown Such trees will have at most 2h leaf nodes and 2h - 1 split nodes

Alternatively you can control the tree size by specifying the number of leaf nodes via the parameter maxleafnodes In this case trees will be grown using bestfirst search where nodes with the highest improve ment in impurity will be expanded first A tree with maxleafnodes k has k - 1 split nodes and thus can model interactions of up to order maxleafnodes - 1

We found that maxleafnodesk gives comparable results to maxdepthk1 but is significantly faster to train at the expense of a slightly higher training error The parameter maxleafnodes corresponds to the variable Jin the chapter on gradient boosting in F2001 and is related to the parameter interactiondepth in R's gbm package where maxleafnodes = interactiondepth - 1

Mathematical formulation

GBRT considers additive models of the following form

$$f(x) = \sum_{j=1}^J h_j(x)$$

where h_j are the basis functions which are usually called weak learners in the context of boosting Gradient Tree Boosting uses decision trees of fixed size as weak learners Decision trees have a number of abilities that make them valuable for boosting namely the ability to handle data of mixed type and the ability to model complex functions Similar to other boosting algorithms GBRT builds the additive model in a greedy fashion

$$f_m(x) = f_{m-1}(x) + h_m(x)$$

where the newly added tree h_m tries to minimize the loss ℓ given the previous ensemble f_{m-1}

$$h_m = \arg \min$$

$$h \sum_{i=1}^n \ell(y_i - f_{m-1}(x_i) + h(x_i))$$

The initial model f_0 is problem specific for leastsquares regression one usually chooses the mean of the target values

Note The initial model can also be specified via the init argument The passed object has to implement fit and predict

Gradient Boosting attempts to solve this minimization problem numerically via steepest descent The steepest descent direction is the negative gradient of the loss function evaluated at the current model \hat{f}_{m-1} which can be calculated for any differentiable loss function

$$\nabla_{\hat{f}_{m-1}} L(y, \hat{f}_{m-1}) = -\sum_{i=1}^n \eta_i \nabla_{\hat{f}_{m-1}} L(y_i, \hat{f}_{m-1})$$

Where the step length η is chosen using line search

$$\eta = \arg \min_{\eta} L(y, \hat{f}_{m-1} + \eta \nabla_{\hat{f}_{m-1}} L(y, \hat{f}_{m-1}))$$

The algorithms for regression and classification only differ in the concrete loss function used

Loss Functions

The following loss functions are supported and can be specified using the parameter loss

- Regression
 - Least squares ls The natural choice for regression due to its superior computational properties The initial model is given by the mean of the target values
 - Least absolute deviation lad A robust loss function for regression The initial model is given by the median of the target values
 - Huber huber Another robust loss function that combines least squares and least absolute deviation use alpha to control the sensitivity with regards to outliers see F2001 for more details
 - Quantile quantile A loss function for quantile regression Use 0 < alpha < 1 to specify the quantile This loss function can be used to create prediction intervals see Prediction Intervals for Gradient Boosting Regression
- Classification
 - Binomial deviance deviance The negative binomial loglikelihood loss function for binary classification provides probability estimates The initial model is given by the log odds ratio
 - Multinomial deviance deviance The negative multinomial loglikelihood loss function for multi class classification with nclasses mutually exclusive classes It provides probability estimates The initial model is given by the prior probability of each class At each iteration nclasses regression trees have to be constructed which makes GBRT rather inefficient for data sets with a large number of classes
 - Exponential loss exponential The same loss function as AdaBoostClassifier Less robust to mislabeled examples than deviance can only be used for binary classification

Regularization

Shrinkage

F2001 proposed a simple regularization strategy that scales the contribution of each weak learner by a factor η

$$\hat{y} = \sum_{t=1}^T \eta_t h_t(x)$$

The parameter η is also called the learning rate because it scales the step length the gradient descent procedure it can be set via the learningrate parameter

The parameter learningrate strongly interacts with the parameter nestimators the number of weak learners to fit Smaller values of learningrate require larger numbers of weak learners to maintain a constant training error Empirical evidence suggests that small values of learningrate favor better test error HTF2009 recommend to set the learning rate to a small constant eg learningrate = 0.1 and choose nestimators by early stopping For a more detailed discussion of the interaction between learningrate and nestimators see R2007

Subsampling

F1999 proposed stochastic gradient boosting which combines gradient boosting with bootstrap averaging bagging At each iteration the base classifier is trained on a fraction subsample of the available training data The subsample is drawn without replacement A typical value of subsample is 0.5

The figure below illustrates the effect of shrinkage and subsampling on the goodness of fit of the model We can clearly see that shrinkage outperforms no shrinkage Subsampling with shrinkage can further increase the accuracy of the model Subsampling without shrinkage on the other hand does poorly

Another strategy to reduce the variance is by subsampling the features analogous to the random splits in RandomForestClassifier The number of subsampled features can be controlled via the maxfeatures parameter

scikitlearn user guide Release 0213

Note Using a small maxfeatures value can significantly decrease the runtime

Stochastic gradient boosting allows to compute outofbag estimates of the test deviance by computing the improvement in deviance on the examples that are not included in the bootstrap sample ie the outofbag examples The improvements are stored in the attribute oobimprovement oobimprovementi holds the improvement in terms of the loss on the OOB samples if you add the ith stage to the current predictions Outofbag estimates can be used for model selection for example to determine the optimal number of iterations OOB estimates are usually very pessimistic thus we recommend to use crossvalidation instead and only use OOB if crossvalidation is too time consuming

Examples

- Gradient Boosting regularization
- Gradient Boosting OutofBag estimates
- OOB Errors for Random Forests

Interpretation

Individual decision trees can be interpreted easily by simply visualizing the tree structure Gradient boosting models however comprise hundreds of regression trees thus they cannot be easily interpreted by visual inspection of the individual trees Fortunately a number of techniques have been proposed to summarize and interpret gradient boosting models

Feature importance

Often features do not contribute equally to predict the target response in many situations the majority of the features are in fact irrelevant When interpreting a model the first question usually is what are those important features and how do they contributing in predicting the target response

Individual decision trees intrinsically perform feature selection by selecting appropriate split points This information can be used to measure the importance of each feature the basic idea is the more often a feature is used in the split points of a tree the more important that feature is This notion of importance can be extended to decision tree ensembles by simply averaging the feature importance of each tree see Feature importance evaluation for more details

The feature importance scores of a fit gradient boosting model can be accessed via the featureimportances

property

```
from sklearndatasets import makehastie102
from sklearnensemble import GradientBoostingClassifier
X y = makehastie102(randomstate=0)
clf = GradientBoostingClassifier(n_estimators=100, learning_rate=10,
    max_depth=1, random_state=0)
fit_X y
clf.feature_importances_
array([0.1, 0.1, 0.1])
```

Examples

scikitlearn user guide Release 0213

•Gradient Boosting regression

Voting Classifier

The idea behind the VotingClassifier is to combine conceptually different machine learning classifiers and use a majority vote or the average predicted probabilities soft vote to predict the class labels Such a classifier can be useful for a set of equally well performing model in order to balance out their individual weaknesses

Majority Class Labels MajorityHard Voting

In majority voting the predicted class label for a particular sample is the class label that represents the majority mode of the class labels predicted by each individual classifier

Eg if the prediction for a given sample is

- classifier 1 class 1
- classifier 2 class 1
- classifier 3 class 2

the VotingClassifier with votinghard would classify the sample as “class 1” based on the majority class label

In the cases of a tie the VotingClassifier will select the class based on the ascending sort order Eg in the following scenario

- classifier 1 class 2
- classifier 2 class 1

the class label 1 will be assigned to the sample

Usage

The following example shows how to fit the majority rule classifier

```
from sklearn import datasets
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import VotingClassifier

iris = datasets.load_iris()
X, y = iris.data[13:], iris.target

clf1 = LogisticRegression(solver='lbfgs', multi_class='multinomial',
                          random_state=1)
clf2 = RandomForestClassifier(n_estimators=50, random_state=1)
clf3 = GaussianNB()
eclf = VotingClassifier(estimators=[('lr', clf1), ('rf', clf2), ('gnb', clf3)],
                        <=>votinghard)

for clf, label in zip([clf1, clf2, clf3, ecclf], ['Logistic Regression', 'Random
Forest', 'naive Bayes', 'Ensemble']):
    <=>Forest naive Bayes Ensemble
31 Supervised learning 303
```

scikitlearn user guide Release 0213

```
scores crossvalscoreclf X y cv5 scoringaccuracy
printAccuracy 02f02f s scoresmean scoresstd
'→label
```

Accuracy 095 004 Logistic Regression
Accuracy 094 004 Random Forest
Accuracy 091 004 naive Bayes
Accuracy 095 004 Ensemble

Weighted Average Probabilities Soft Voting

In contrast to majority voting hard voting soft voting returns the class label as argmax of the sum of predicted probabilities

Specific weights can be assigned to each classifier via the weights parameter When weights are provided the predicted class probabilities for each classifier are collected multiplied by the classifier weight and averaged The final class label is then derived from the class label with the highest average probability

To illustrate this with a simple example let's assume we have 3 classifiers and a 3class classification problems where we assign equal weights to all classifiers w11 w21 w31

The weighted average probabilities for a sample would then be calculated as follows

```
classifier class 1 class 2 class 3
classifier 1 w1 02 w1 05 w1 03
classifier 2 w2 06 w2 03 w2 01
classifier 3 w3 03 w3 04 w3 03
weighted average 037 04 023
```

Here the predicted class label is 2 since it has the highest average probability

The following example illustrates how the decision regions may change when a soft VotingClassifier is used based on an linear Support Vector Machine a Decision Tree and a Knearest neighbor classifier

```
from sklearn import datasets
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC (Support Vector Classification)
from itertools import product
from sklearn.ensemble import VotingClassifier
```

Loading some example data

```
iris datasetsloadiris
```

```
X irisdata 0 2
```

```
y iristarget
```

Training classifiers

```
clf1 DecisionTreeClassifiermaxdepth4
```

```
clf2 KNeighborsClassifiernneighbors7
```

```
clf3 SVCgammascale kernelrbf probability True
```

```
ecf VotingClassifierestimatorsdt clf1 knn clf2 svc clf3
```

```
votingsoft weights2 1 2
```

```
clf1 clf1fitX y
```

```
clf2 clf2fitX y
```

```
clf3 clf3fitX y
```

```
ecf eclffitX y
```


scikitlearn user guide Release 0213

Using the VotingClassifier with GridSearchCV

The VotingClassifier can also be used together with GridSearchCV in order to tune the hyperparameters of the individual estimators

```
from sklearn.model_selection import GridSearchCV
clf1 = LogisticRegressionSolver(bfgs, multi_class='multinomial',
                               random_state=1)
clf2 = RandomForestClassifier(random_state=1)
clf3 = GaussianNB()
eclf = VotingClassifier(estimators=[clf1, clf2, clf3], voting='soft')
params = {'lr': 0.01, 'n_estimators': 20, 'max_depth': 200}
grid = GridSearchCV(estimator=eclf, param_grid=params, cv=5)
grid.fit(X_train, y_train)
```

Usage
In order to predict the class labels based on the predicted class probabilities, scikitlearn estimators in the VotingClassifier must support the predict_proba method

```
eclf = VotingClassifier(estimators=[clf1, clf2, clf3], voting='soft')
```

Optionally, weights can be provided for the individual classifiers

```
eclf = VotingClassifier(estimators=[clf1, clf2, clf3], voting='soft', weights=[2, 5, 1])
```

VotingRegressor

The idea behind the VotingRegressor is to combine conceptually different machine learning regressors and return the average predicted values. Such a regressor can be useful for a set of equally well performing models in order to balance out their individual weaknesses.

The following example shows how to fit the VotingRegressor

```
from sklearn import datasets
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import VotingRegressor
```

Loading some example data

```
boston = datasets.load_boston()
```

X = boston.data

y = boston.target

Training classifiers

```
reg1 = GradientBoostingRegressor(random_state=1, n_estimators=10)
```

```
reg2 = RandomForestRegressor(random_state=1, n_estimators=10)
```

```
reg3 = LinearRegression()
```


scikitlearn user guide Release 0213

ereg VotingRegressorestimatorsrgb reg1 rf reg2 lr reg3

ereg eregfitX y

Examples

- Plot individual and voting regression predictions

3112 Multiclass and multilabel algorithms

Warning All classifiers in scikitlearn do multiclass classification outofthebox You don't need to use the

sklearnmulticlass module unless you want to experiment with different multiclass strategies

The sklearnmulticlass module implements metaestimators to solvemulticlass andmultilabel clas

sification problems by decomposing such problems into binary classification problems Multitarget regression is also supported

- Multiclass classification means a classification task with more than two classes eg classify a set of images of fruits which may be oranges apples or pears Multiclass classification makes the assumption that each sample is assigned to one and only one label a fruit can be either an apple or a pear but not both at the same time

- Multilabel classification assigns to each sample a set of target labels This can be thought as predicting proper ties of a datapoint that are not mutually exclusive such as topics that are relevant for a document A text might be about any of religion politics finance or education at the same time or none of these

- Multioutput regression assigns each sample a set of target values This can be thought of as predicting several properties for each datapoint such as wind direction and magnitude at a certain location

31 Supervised learning 307

scikitlearn user guide Release 0213

•Multioutputmulticlass classification andmultitask classification means that a single estimator has to handle several joint classification tasks This is both a generalization of the multilabel classification task which only considers binary classification as well as a generalization of the multiclass classification task The output format is a 2d numpy array or sparse matrix

The set of labels can be different for each output variable For instance a sample could be assigned “pear” for an output variable that takes possible values in a finite set of species such as “pear” “apple” and “blue” or “green” for a second output variable that takes possible values in a finite set of colors such as “green” “red” “blue” “yellow”

This means that any classifiers handling multioutput multiclass or multitask classification tasks support the multilabel classification task as a special case Multitask classification is similar to the multioutput classification task with different model formulations For more information see the relevant estimator documentation All scikitlearn classifiers are capable of multiclass classification but the metaestimators offered by sklearn multiclass permit changing the way they handle more than two classes because this may have an effect on classifier performance either in terms of generalization error or required computational resources Below is a summary of the classifiers supported by scikitlearn grouped by strategy you don’t need the metaestimators in this class if you’re using one of these unless you want custom multiclass behavior

- Inherently multiclass
 - sklearnnaivebayesBernoulliNB
 - sklearnDecisionTreeClassifier
 - sklearnExtraTreeClassifier
 - sklearnensembleExtraTreesClassifier
 - sklearnnaivebayesGaussianNB
 - sklearnneighborsKNeighborsClassifier
 - sklearnsemisupervisedLabelPropagation
 - sklearnsemisupervisedLabelSpreading
 - sklearnDiscriminantAnalysisLinearDiscriminantAnalysis
 - sklearnsvmLinearSVC setting multiclass”crammersinger”
 - sklearnlinearModelLogisticRegression setting multiclass”multinomial”
 - sklearnlinearModelLogisticRegressionCV setting multiclass”multinomial”
 - sklearnneuralnetworkMLPClassifier
 - sklearnneighborsNearestCentroid
 - sklearnDiscriminantAnalysisQuadraticDiscriminantAnalysis
 - sklearnneighborsRadiusNeighborsClassifier
 - sklearnensembleRandomForestClassifier
 - sklearnlinearModelRidgeClassifier
 - sklearnlinearModelRidgeClassifierCV
 - Multiclass as OneVsOne
 - sklearnsvmNuSVC
 - sklearnsvmSVC
- 308 Chapter 3 User Guide

scikitlearn user guide Release 0213

-sklearnGaussianProcessGaussianProcessClassifier setting multiclass
"onevsone"

•Multiclass as OneVsAll

-sklearnensembleGradientBoostingClassifier

-sklearnGaussianProcessGaussianProcessClassifier setting multiclass
"onevsrest"

-sklearnsvmLinearSVC setting multiclass"ovr"

-sklearnlinearmodelLogisticRegression setting multiclass"ovr"

-sklearnlinearmodelLogisticRegressionCV setting multiclass"ovr"

-sklearnlinearmodelSGDClassifier

-sklearnlinearmodelPerceptron

-sklearnlinearmodelPassiveAggressiveClassifier

•Support multilabel

-sklearntreeDecisionTreeClassifier

-sklearntreeExtraTreeClassifier

-sklearnensembleExtraTreesClassifier

-sklearnneighborsKNeighborsClassifier

-sklearnneuralnetworkMLPClassifier

-sklearnneighborsRadiusNeighborsClassifier

-sklearnensembleRandomForestClassifier

-sklearnlinearmodelRidgeClassifierCV

•Support multiclassmultioutput

-sklearntreeDecisionTreeClassifier

-sklearntreeExtraTreeClassifier

-sklearnensembleExtraTreesClassifier

-sklearnneighborsKNeighborsClassifier

-sklearnneighborsRadiusNeighborsClassifier

-sklearnensembleRandomForestClassifier

Warning At present no metric in sklearnmetrics supports the multioutputmulticlass classification task

Multilabel classification format

In multilabel learning the joint set of binary classification tasks is expressed with label binary indicator array each sample is one row of a 2d array of shape nsamples nclasses with binary values the one ie the non zero elements corresponds to the subset of labels An array such as nparray1 0 0 0 1 1 0 0 0

represents label 0 in the first sample labels 1 and 2 in the second sample and no labels in the third sample

Producing multilabel data as a list of sets of labels may be more intuitive The MultiLabelBinarizer transformer

can be used to convert between a collection of collections of labels and the indicator format

31 Supervised learning 309

scikitlearn user guide Release 0213

- “Solving multiclass learning problems via errorcorrecting output codes” Dietterich T Bakiri G Journal of Artificial Intelligence Research 2 1995
- “The Elements of Statistical Learning” Hastie T Tibshirani R Friedman J page 606 secondedition 2008

Multioutput regression

Multioutput regression support can be added to any regressor with MultiOutputRegressor. This strategy consists of fitting one regressor per target. Since each target is represented by exactly one regressor, it is possible to gain knowledge about the target by inspecting its corresponding regressor. As MultiOutputRegressor fits one regressor per target, it can not take advantage of correlations between targets.

Below is an example of multioutput regression:

```
from sklearn.datasets import make_regression
from sklearn.multioutput import MultiOutputRegressor
from sklearn.ensemble import GradientBoostingRegressor
X, y = make_regression(n_samples=10, n_targets=3, random_state=1)
MultiOutputRegressor(GradientBoostingRegressor()).fit(X, y)
#> predict(X)
array([15475474165, 14703498585, 5003812219,
       712165031, 512914884, 8146081961,
       1878948621, 10044373091, 1388978285,
       14162745778, 9502891072, 19148204257,
       9703260883, 16534867495, 13952003279,
       12392529176, 2125719016, 784253,
       12225193977, 8516443186, 10712274212,
       30170388, 9480956739, 1216979946,
       14072667194, 17650941682, 1750447799,
       14937967282, 8115699552, 572850319])
```

Multioutput classification

Multioutput classification support can be added to any classifier with MultiOutputClassifier. This strategy consists of fitting one classifier per target. This allows multiple target variable classifications. The purpose of this class is to extend estimators to be able to estimate a series of target functions $f_1, f_2, f_3, \dots, f_n$ that are trained on a single X predictor matrix to predict a series of responses $y_1, y_2, y_3, \dots, y_n$.

Below is an example of multioutput classification:

```
from sklearn.datasets import make_classification
from sklearn.multioutput import MultiOutputClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.utils import shuffle
import numpy as np
X, y1 = make_classification(n_samples=10, n_features=100, n_informative=30, n
#> classes=3, random_state=1)
y2 = shuffle(y1, random_state=1)
y3 = shuffle(y1, random_state=2)
Y = np.vstack((y1, y2, y3))
n_samples, n_features = X.shape
n_outputs = Y.shape[1]
n_classes = 3
forest = RandomForestClassifier(n_estimators=100, random_state=1)
MultiOutputClassifier(forest).fit(X, Y)
```

scikitlearn user guide Release 0213

multitargetforestfitX YpredictX

array2 2 0

1 2 1

2 1 0

0 0 2

0 2 1

0 0 2

1 1 0

1 1 1

0 0 2

2 0 0

Classifier Chain

Classifier chains see ClassifierChain are a way of combining a number of binary classifiers into a single multilabel model that is capable of exploiting correlations among targets

For a multilabel classification problem with N classes N binary classifiers are assigned an integer between 0 and N1

These integers define the order of models in the chain Each classifier is then fit on the available training data plus the true labels of the classes whose models were assigned a lower number

When predicting the true labels will not be available Instead the predictions of each model are passed on to the subsequent models in the chain to be used as features

Clearly the order of the chain is important The first model in the chain has no information about the other labels while the last model in the chain has features indicating the presence of all of the other labels In general one does not know the optimal ordering of the models in the chain so typically many randomly ordered chains are fit and their predictions are averaged together

References

Jesse Read Bernhard Pfahringer Geoff Holmes Eibe Frank “Classifier Chains for Multilabel Classification” 2009

Regressor Chain

Regressor chains see RegressorChain is analogous to ClassifierChain as a way of combining a number of regressions into a single multitarget model that is capable of exploiting correlations among targets

3113 Feature selection

The classes in the sklearnfeatureselection module can be used for feature selectiondimensionality reduction on sample sets either to improve estimators’ accuracy scores or to boost their performance on very high dimensional datasets

Removing features with low variance

VarianceThreshold is a simple baseline approach to feature selection It removes all features whose variance doesn’t meet some threshold By default it removes all zero variance features ie features that have the same value in all samples

scikitlearn user guide Release 0213

As an example suppose that we have a dataset with boolean features and we want to remove all features that are either one or zero on or off in more than 80 of the samples Boolean features are Bernoulli random variables and the variance of such variables is given by

```
Var = p(1-p)
so we can select using the threshold 81 / 8
from sklearn.feature_selection import VarianceThreshold
X = [[0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1]
sel = VarianceThreshold(threshold=0.8)
sel.fit_transform(X)
array([[0, 1,
        1, 0,
        0, 0,
        1, 1,
        1, 0,
        1, 1]
```

As expected VarianceThreshold has removed the first column which has a probability of 56.8% of containing a zero

Univariate feature selection

Univariate feature selection works by selecting the best features based on univariate statistical tests It can be seen as a preprocessing step to an estimator Scikitlearn exposes feature selection routines as objects that implement the transform method

- SelectKBest removes all but the k highest scoring features
- SelectPercentile removes all but a user specified highest scoring percentage of features
- using common univariate statistical tests for each feature false positive rate SelectFpr false discovery rate SelectFdr or family wise error SelectFwe
- GenericUnivariateSelect allows to perform univariate feature selection with a configurable strategy This allows to select the best univariate selection strategy with hyperparameter search estimator

For instance we can perform a chi2 test to the samples to retrieve only the two best features as follows

```
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
iris = load_iris()
X, y = iris.data, iris.target
X.shape
(150, 4)
X_new = SelectKBest(chi2, k=2).fit_transform(X, y)
X_new.shape
(150, 2)
```

These objects take as input a scoring function that returns univariate scores and p-values or only scores for

- SelectKBest and SelectPercentile
- For regression f_regression mutual_info_regression
- For classification chi2 f_classif mutual_info_classif

scikitlearn user guide Release 0213

The methods based on Ftest estimate the degree of linear dependency between two random variables On the other hand mutual information methods can capture any kind of statistical dependency but being nonparametric they require more samples for accurate estimation

Feature selection with sparse data

If you use sparse data ie data represented as sparse matrices chi2 mutualinfo regression

mutualinfoclassif will deal with the data without making it dense

Warning Beware not to use a regression scoring function with a classification problem you will get useless results

Examples

- Univariate Feature Selection
 - Comparison of Ftest and mutual information
- Recursive feature elimination

Given an external estimator that assigns weights to features eg the coefficients of a linear model recursive feature elimination RFE is to select features by recursively considering smaller and smaller sets of features First the estimator is trained on the initial set of features and the importance of each feature is obtained either through a coef attribute or through a featureimportances attribute Then the least important features are pruned from current set of features That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached

RFECV performs RFE in a crossvalidation loop to find the optimal number of features

Examples

- Recursive feature elimination A recursive feature elimination example showing the relevance of pixels in a digit classification task
- Recursive feature elimination with crossvalidation A recursive feature elimination example with automatic tuning of the number of features selected with crossvalidation

Feature selection using SelectFromModel

SelectFromModel is a metatransformer that can be used along with any estimator that has a coef or featureimportances attribute after fitting The features are considered unimportant and removed if the corresponding coef or featureimportances values are below the provided threshold parameter Apart from specifying the threshold numerically there are builtin heuristics for finding a threshold using a string argument Available heuristics are “mean” “median” and float multiples of these like “01mean”

For examples on how it is to be used refer to the sections below

316 Chapter 3 User Guide

scikitlearn user guide Release 0213

Examples

•Feature selection using SelectFromModel and LassoCV Selecting the two most important features from the Boston dataset without knowing the threshold beforehand

L1based feature selection

Linear models penalized with the L1 norm have sparse solutions many of their estimated coefficients are zero When the goal is to reduce the dimensionality of the data to use with another classifier they can be used along withfeatureselectionSelectFromModel to select the nonzero coefficients In particular sparse estimators useful for this purpose are the linearmodelLasso for regression and of linearmodel

LogisticRegression andsvmLinearSVC for classification

```
from sklearnsvm import LinearSVC
from sklearndatasets import loadiris
from sklearnfeatureselection import SelectFromModel
```

```
iris = loadiris
X, y = iris.data, iris.target
X.shape
150, 4
```

```
lsvc = LinearSVCC001 penaltyl1 dual FalsefitX, y
model = SelectFromModel(lsvc, prefit=True)
```

```
X_new = model.transform(X)
X_new.shape
150, 3
```

With SVMs and logisticregression the parameter C controls the sparsity the smaller C the fewer features selected With Lasso the higher the alpha parameter the fewer features selected

Examples

•Classification of text documents using sparse features Comparison of different algorithms for document classification including L1based feature selection

L1recovery and compressive sensing

For a good choice of alpha the Lasso can fully recover the exact set of nonzero variables using only few observations provided certain specific conditions are met In particular the number of samples should be “sufficiently large” or L1 models will perform at random where “sufficiently large” depends on the number of nonzero coefficients the logarithm of the number of features the amount of noise the smallest absolute value of nonzero coefficients and the structure of the design matrix X In addition the design matrix must display certain specific properties such as not being too correlated

There is no general rule to select an alpha parameter for recovery of nonzero coefficients It can be set by cross validation LassoCV orLassoLarsCV though this may lead to underpenalized models including a small number of nonrelevant variables is not detrimental to prediction score BIC LassoLarsIC tends on the opposite to set high values of alpha

Reference Richard G Baraniuk “Compressive Sensing” IEEE Signal Processing Magazine 120 July 2007 <http://users.isr.ist.utl.pt/guarCSnotes.pdf>

31 Supervised learning 317

scikitlearn user guide Release 0213

Treebased feature selection

Treebased estimators see the sklearnntree module and forest of trees in the sklearnensemble module can be used to compute feature importances which in turn can be used to discard irrelevant features when coupled with thesklearnfeatureselectionSelectFromModel metatransformer

```
from sklearnensemble import ExtraTreesClassifier
from sklearnndatasets import loadiris
from sklearnfeatureselection import SelectFromModel
iris = loadiris
```

```
X, y = iris.data, iris.target
X.shape
(150, 4)
clf = ExtraTreesClassifier(n_estimators=50)
clf.fit(X, y)
clf.feature_importances_
array([0.04, 0.05, 0.4, 0.4])
model = SelectFromModel(clf, prefit=True)
X_new = model.transform(X)
X_new.shape
(150, 2)
```

- Examples
- Feature importances with forests of trees example on synthetic data showing the recovery of the actually meaningful features
 - Pixel importances with a parallel forest of trees example on face recognition data

Feature selection as part of a pipeline

Feature selection is usually used as a preprocessing step before doing the actual learning The recommended way to do this in scikitlearn is to use a sklearnpipelinePipeline

```
clf = Pipeline([
    ('featureselection', SelectFromModel(LinearSVC(penalty=l1))),
    ('classification', RandomForestClassifier())])
```

clf.fit(X, y)

In this snippet we make use of a sklearnsvmLinearSVC coupled with sklearnfeatureselection SelectFromModel to evaluate feature importances and select the most relevant features Then a sklearn ensembleRandomForestClassifier is trained on the transformed output ie using only relevant features

You can perform similar operations with the other feature selection methods and also classifiers that provide a way to evaluate feature importances of course See the sklearnpipelinePipeline examples for more details

3114 SemiSupervised

Semisupervised learning is a situation in which in your training data some of the samples are not labeled The semi supervised estimators in sklearnsemisupervised are able to make use of this additional unlabeled data to better capture the shape of the underlying data distribution and generalize better to new samples These algorithms can perform well when we have a very small amount of labeled points and a large amount of unlabeled points

Unlabeled entries in y

It is important to assign an identifier to unlabeled points along with the labeled data when training the model with thefit method The identifier that this implementation uses is the integer value -1

Label Propagation

Label propagation denotes a few variations of semisupervised graph inference algorithms

A few features available in this model

- Can be used for classification and regression tasks
- Kernel methods to project data into alternate dimensional spaces

scikitlearn provides two label propagation models LabelPropagation andLabelSpreading Both work by constructing a similarity graph over all items in the input dataset

Fig 31 An illustration of labelpropagation the structure of unlabeled observations is consistent with the class structure and thus the class label can be propagated to the unlabeled observations of the training set LabelPropagation andLabelSpreading differ in modifications to the similarity matrix that graph and the clamping effect on the label distributions Clamping allows the algorithm to change the weight of the true ground labeled data to some degree The LabelPropagation algorithm performs hard clamping of input labels which means $\gamma = 0$ This clamping factor can be relaxed to say $\gamma = 0.2$ which means that we will always retain 80 percent of our original label distribution but the algorithm gets to change its confidence of the distribution within 20 percent LabelPropagation uses the raw similarity matrix constructed from the data with no modifications In contrast LabelSpreading minimizes a loss function that has regularization properties as such it is often more robust to noise The algorithm iterates on a modified version of the original graph and normalizes the edge weights by computing the normalized graph Laplacian matrix This procedure is also used in Spectral clustering Label propagation models have two builtin kernel methods Choice of kernel effects both scalability and performance of the algorithms The following are available

- rbf $\exp(-\gamma \|x - x'\|^2)$ γ is specified by keyword gamma
- knn $\frac{1}{\|x - x'\|}$ γ is specified by keyword nneighbors

31 Supervised learning 319

scikitlearn user guide Release 0213

The RBF (Radial Basis Function) kernel will produce a fully connected graph which is represented in memory by a dense matrix This matrix may be very large and combined with the cost of performing a full matrix multiplication calculation for each iteration of the algorithm can lead to prohibitively long running times On the other hand the KNN kernel will produce a much more memoryfriendly sparse matrix which can drastically reduce running times

Examples

- Decision boundary of label propagation versus SVM (Support Vector Machine) on the Iris dataset
- Label Propagation learning a complex structure
- Label Propagation digits Demonstrating performance
- Label Propagation digits active learning

References

1 Yoshua Bengio Olivier Delalleau Nicolas Le Roux In SemiSupervised Learning 2006 pp 193216
2 Olivier Delalleau Yoshua Bengio Nicolas Le Roux Efficient NonParametric Function Induction in Semi Supervised Learning AISTAT 2005 <https://research.microsoft.com/en-us/people/nicolas/efficientsslp.pdf>

3115 Isotonic regression

The classIsotonicRegression fits a nondecreasing function to data It solves the following problem

minimize \sum

$\sum_{i=1}^n (y_i - \hat{y}_i)^2$

subject to $\hat{y}_1 \leq \hat{y}_2 \leq \dots \leq \hat{y}_n$

where each y_i is strictly positive and each \hat{y}_i is an arbitrary real number It yields the vector which is composed of nondecreasing elements the closest in terms of mean squared error In practice this list of elements forms a function that is piecewise linear

3116 Probability calibration

When performing classification you often want not only to predict the class label but also obtain a probability of the respective label This probability gives you some kind of confidence on the prediction Some models can give you poor estimates of the class probabilities and some even do not support probability prediction The calibration module allows you to better calibrate the probabilities of a given model or to add support for probability prediction Well calibrated classifiers are probabilistic classifiers for which the output of the predict_proba method can be directly interpreted as a confidence level For instance a well calibrated binary classifier should classify the samples such that among the samples to which it gave a predict_proba value close to 0.8 approximately 80 actually belong to the positive class The following plot compares how well the probabilistic predictions of different classifiers are calibrated LogisticRegression returns well calibrated predictions by default as it directly optimizes logloss In contrast the other methods return biased probabilities with different biases per method

- GaussianNB tends to push probabilities to 0 or 1 note the counts in the histograms This is mainly because it makes the assumption that features are conditionally independent given the class which is not the case in this dataset which contains 2 redundant features

scikitlearn user guide Release 0213

•RandomForestClassifier shows the opposite behavior the histograms show peaks at approximately 0.2 and 0.9 probability while probabilities close to 0 or 1 are very rare An explanation for this is given by NiculescuMizil and Caruana4 “Methods such as bagging and random forests that average predictions from a base set of models can have difficulty making predictions near 0 and 1 because variance in the underlying base models will bias predictions that should be near zero or one away from these values Because predictions are restricted to the interval [0,1] errors caused by variance tend to be onesided near zero and one For example if a model should predict p = 0 for a case the only way bagging can achieve this is if all bagged trees predict zero If we add noise to the trees that bagging is averaging over this noise will cause some trees to predict values larger than 0 for this case thus moving the average prediction of the bagged ensemble away from 0 We observe this effect most strongly with random forests because the baselevel trees trained with random forests have relatively high variance due to feature subsetting” As a result the calibration curve also referred to as the reliability diagram Wilks 19955 shows a characteristic sigmoid shape indicating that the classifier could trust its “intuition” more and return probabilities closer to 0 or 1 typically

• Linear Support Vector Classification LinearSVC shows an even more sigmoid curve as the RandomForest Classifier which is typical for maximummargin methods compare NiculescuMizil and Caruana4 which focus on hard samples that are close to the decision boundary the support vectors

Two approaches for performing calibration of probabilistic predictions are provided a parametric approach based on Platt’s sigmoid model and a nonparametric approach based on isotonic regression sklearnisotonic Probability calibration should be done on new data not used for model fitting The class CalibratedClassifierCV uses a crossvalidation generator and estimates for each split the model parameter on the train samples and the calibration of the test samples The probabilities predicted for the folds are then averaged Already fitted classifiers can be calibrated by CalibratedClassifierCV via the parameter cv=”prefit” In this case the user has to take care manually that data for model fitting and calibration are disjoint

The following images demonstrate the benefit of probability calibration The first image present a dataset with 2 classes and 3 blobs of data The blob in the middle contains random samples of each class The probability for the samples in this blob should be 0.5

The following image shows on the data above the estimated probability using a Gaussian naive Bayes classifier without calibration with a sigmoid calibration and with a nonparametric isotonic calibration One can observe that the non parametric model provides the most accurate probability estimates for samples in the middle ie 0.5

The following experiment is performed on an artificial dataset for binary classification with 100000 samples 1000 of them are used for model fitting with 20 features Of the 20 features only 2 are informative and 10 are redundant The figure shows the estimated probabilities obtained with logistic regression a linear supportvector classifier SVC (Support Vector Classification) and linear SVC (Support Vector Classification) with both isotonic calibration and sigmoid calibration The Brier score is a metric which is a combination of calibration loss and refinement loss brierscoreloss reported in the legend the smaller the better Calibration loss is defined as the mean squared deviation from empirical probabilities derived from the slope of ROC segments Refinement loss can be defined as the expected optimal loss as measured by the area under the optimal cost curve

One can observe here that logistic regression is well calibrated as its curve is nearly diagonal Linear SVC (Support Vector Classification)’s calibration curve or reliability diagram has a sigmoid curve which is typical for an underconfident classifier In the case of LinearSVC this is caused by the margin property of the hinge loss which lets the model focus on hard samples that are close to the decision boundary the support vectors Both kinds of calibration can fix this issue and yield nearly identical results The next figure shows the calibration curve of Gaussian naive Bayes on the same data with both kinds of calibration and also without calibration

One can see that Gaussian naive Bayes performs very badly but does so in an other way than linear SVC (Support Vector Classification) While linear SVC (Support Vector Classification) exhibited a sigmoid calibration curve Gaussian naive Bayes’ calibration curve has a transposedsigmoid shape This is typical for an overconfident classifier In this case the classifier’s overconfidence is caused by the redundant features which violate the naive Bayes assumption of featureindependence

4Predicting Good Probabilities with Supervised Learning A NiculescuMizil R Caruana ICML 2005

5On the combination of forecast probabilities for consecutive precipitation periods Wea Forecasting 5 640–650 Wilks D S 1990a

31 Supervised learning 323

Calibration of the probabilities of Gaussian naive Bayes with isotonic regression can fix this issue as can be seen from the nearly diagonal calibration curve Sigmoid calibration also improves the brier score slightly albeit not as strongly as the nonparametric isotonic calibration This is an intrinsic limitation of sigmoid calibration whose parametric form assumes a sigmoid rather than a transposedsigmoid curve The nonparametric isotonic calibration model however makes no such strong assumptions and can deal with either shape provided that there is sufficient calibration data In general sigmoid calibration is preferable in cases where the calibration curve is sigmoid and where there is limited calibration data while isotonic calibration is preferable for nonsigmoid calibration curves and in situations where large amounts of data are available for calibration

CalibratedClassifierCV can also deal with classification tasks that involve more than two classes if the base estimator can do so In this case the classifier is calibrated first for each class separately in an onevsrest fashion When predicting probabilities for unseen data the calibrated probabilities for each class are predicted separately As those probabilities do not necessarily sum to one a postprocessing is performed to normalize them The next image illustrates how sigmoid calibration changes predicted probabilities for a 3class classification problem Illustrated is the standard 2simplex where the three corners correspond to the three classes Arrows point from the probability vectors predicted by an uncalibrated classifier to the probability vectors predicted by the same classifier after sigmoid calibration on a holdout validation set Colors indicate the true class of an instance red class 1 green class 2 blue class 3

The base classifier is a random forest classifier with 25 base estimators trees If this classifier is trained on all 800 training datapoints it is overly confident in its predictions and thus incurs a large logloss Calibrating an identical classifier which was trained on 600 datapoints with method'sigmoid' on the remaining 200 datapoints reduces the confidence of the predictions ie moves the probability vectors from the edges of the simplex towards the center

scikitlearn user guide Release 0213

This calibration results in a lower logloss Note that an alternative would have been to increase the number of base estimators which would have resulted in a similar decrease in logloss

References

- Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers B Zadrozny C Elkan ICML 2001
- Transforming Classifier Scores into Accurate Multiclass Probability Estimates B Zadrozny C Elkan KDD 2002
- Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods J Platt 1999

3117 Neural network models supervised

Warning This implementation is not intended for largescale applications In particular scikitlearn offers no GPU support For much faster GPUbased implementations as well as frameworks offering much more flexibility to build deep learning architectures see Related Projects

Multilayer Perceptron

Multilayer Perceptron MLP is a supervised learning algorithm that learns a function $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$ by training on a dataset where n is the number of dimensions for input and m is the number of dimensions for output Given a set of features x_1, x_2, \dots, x_n and a target y it can learn a nonlinear function approximator for either classification or regression It is different from logistic regression in that between the input and the output layer there can be one or more nonlinear layers called hidden layers Figure 1 shows a one hidden layer MLP with scalar output

Fig 32 Figure 1 One hidden layer MLP

The leftmost layer known as the input layer consists of a set of neurons x_1, x_2, \dots, x_n representing the input features Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation $w_1x_1 + w_2x_2 + \dots + w_nx_n$ followed by a nonlinear activation function $f: \mathbb{R} \rightarrow \mathbb{R}$ like the hyperbolic tan function The output layer receives the values from the last hidden layer and transforms them into output values

scikitlearn user guide Release 0213

The module contains the public attributes `coefs` and `intercepts` `coefs` is a list of weight matrices where weight matrix at index `i` represents the weights between layer `i` and layer `i+1` `intercepts` is a list of bias vectors where the vector at index `i` represents the bias values added to layer `i`

The advantages of Multilayer Perceptron are

- Capability to learn nonlinear models
- Capability to learn models in realtime online learning using `partialfit`

The disadvantages of Multilayer Perceptron MLP include

- MLP with hidden layers have a nonconvex loss function where there exists more than one local minimum Therefore different random weight initializations can lead to different validation accuracy
- MLP requires tuning a number of hyperparameters such as the number of hidden neurons layers and iterations
- MLP is sensitive to feature scaling

Please see Tips on Practical Use section that addresses some of these disadvantages

Classification

`ClassMLPClassifier` implements a multilayer perceptron MLP algorithm that trains using Backpropagation MLP trains on two arrays `array X` of size `nsamples nfeatures` which holds the training samples represented as floating point feature vectors and `array y` of size `nsamples` which holds the target values class labels for the training samples

```
from sklearn.neuralnetwork import MLPClassifier
```

```
X = [[0, 0, 1, 1],
```

```
     [0, 1, 0, 1]]
```

```
y = [0, 1]
```

```
clf = MLPClassifier(solver='lbfgs', alpha=1e-5,
```

```
                   hidden_layer_sizes=(5, 2), random_state=1)
```

```
clf.fit(X, y)
```

```
MLPClassifier(activation=relu, alpha=1e-05, batch_size=auto,
```

```
beta=109, beta_2=0.999, early_stopping=False,
```

```
epsilon=1e-08, hidden_layer_sizes=(5, 2),
```

```
learning_rate=constant, learning_rate_init=0.001,
```

```
max_iter=200, momentum=0.9, n_iter_no_change=10,
```

```
nesterovs_momentum=True, power_t=0.5, random_state=1,
```

```
shuffle=True, solver='lbfgs', tol=0.0001,
```

```
validation_fraction=0.1, verbose=False, warm_start=False)
```

After fitting training the model can predict labels for new samples

```
clf.predict([[2, 2, 1, 2],
```

```
            [1, 0]])
```

MLP can fit a nonlinear model to the training data `clf.coefs` contains the weight matrices that constitute the model parameters

```
coef_shape = [coefs[i].shape for i in range(len(coefs))]
```

```
[2, 5, 5, 2, 2, 1]
```

Currently `MLPClassifier` supports only the CrossEntropy loss function which allows probability estimates by running the `predict_proba` method

31 Supervised learning 331

scikitlearn user guide Release 0213

MLP trains using Backpropagation More precisely it trains using some form of gradient descent and the gradients are calculated using Backpropagation For classification it minimizes the CrossEntropy loss function giving a vector of probability estimates [] per sample []

```
clf.predict_proba(2, 1, 2)
array([1.967e+04, 9.998e+01,
       1.967e+04, 9.998e+01])
```

MLPClassifier supports multiclass classification by applying Softmax as the output function Further the model supports multilabel classification in which a sample can belong to more than one class For each class the raw output passes through the logistic function Values larger or equal to 0.5 are rounded to 1 otherwise to 0 For a predicted output of a sample the indices where the value is 1 represents the assigned classes of that sample

```
X = [[0, 1, 1],
     [0, 1, 1],
     [0, 1, 1]]
y = [0, 1, 1, 1]
clf = MLPClassifier(solver='lbfgs', alpha=1e-5,
                   hidden_layer_sizes=(15, ), random_state=1)
```

```
clf.fit(X, y)
MLPClassifier(activation=relu, alpha=1e-05, batch_size='auto',
              beta_1=0.9, beta_2=0.999, early_stopping=False,
              epsilon=1e-08, hidden_layer_sizes=(15, ),
              learning_rate='constant', learning_rate_init=0.001,
              max_iter=200, momentum=0.9, n_iter_no_change=10,
              nesterovs_momentum=True, power_t=0.5, random_state=1,
              shuffle=True, solver='lbfgs', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False)
clf.predict(1, 2)
array([1, 1])
clf.predict(0, 0)
array([0, 1])
```

See the examples below and the docstring of MLPClassifier.fit for further information

Examples

- Compare Stochastic learning strategies for MLPClassifier
- Visualization of MLP weights on MNIST

Regression

ClassMLPRegressor implements a multilayer perceptron MLP that trains using backpropagation with no activation function in the output layer which can also be seen as using the identity function as activation function Therefore it uses the square error as the loss function and the output is a set of continuous values

MLPRegressor also supports multioutput regression in which a sample can have more than one target

Regularization

BothMLPRegressor andMLPClassifier use parameter alpha for regularization L2 regularization term which helps in avoiding overfitting by penalizing weights with large magnitudes Following plot displays varying decision function with value of alpha

scikitlearn user guide Release 0213
See the examples below for further information
Examples
•Varying regularization in Multilayer Perceptron

Algorithms
MLP trains using Stochastic Gradient Descent Adam or LBFGS Stochastic Gradient Descent SGD updates pa
rameters using the gradient of the loss function with respect to a parameter that needs adaptation ie
 $\theta \leftarrow \theta - \eta \nabla_{\theta} L(\theta)$
where η is the learning rate which controls the stepsize in the parameter space search $L(\theta)$ is the loss function used
for the network

More details can be found in the documentation of SGD
Adam is similar to SGD in a sense that it is a stochastic optimizer but it can automatically adjust the amount to update
parameters based on adaptive estimates of lowerorder moments
With SGD or Adam training supports online and minibatch learning
LBFGS is a solver that approximates the Hessian matrix which represents the secondorder partial derivative of a
function Further it approximates the inverse of the Hessian matrix to perform parameter updates The implementation
uses the Scipy version of LBFGS
If the selected solver is 'LBFGS' training does not support online nor minibatch learning
31 Supervised learning 333

Complexity

Suppose there are n training samples m features h hidden layers each containing h neurons for simplicity and o output neurons The time complexity of backpropagation is $O(n \cdot m \cdot h \cdot o \cdot i)$ where i is the number of iterations Since backpropagation has a high time complexity it is advisable to start with smaller number of hidden neurons and few hidden layers for training

Mathematical formulation

Given a set of training examples x_1, x_2, \dots, x_n where $x_i \in \mathbb{R}^m$ and $y_i \in \{0, 1\}$ a one hidden layer one hidden neuron MLP learns the function $f(x) = \sigma(w_1 x + b_1)$

where $w_1 \in \mathbb{R}^{m \times h}$ and $b_1 \in \mathbb{R}^h$ are model parameters w_1 and b_1 represent the weights of the input layer and hidden layer respectively and $w_2 \in \mathbb{R}^{h \times o}$ and $b_2 \in \mathbb{R}^o$ represent the bias added to the hidden layer and the output layer respectively $\sigma: \mathbb{R} \rightarrow \mathbb{R}$ is the activation function set by default as the hyperbolic tan It is given as

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

For binary classification $\sigma(x)$ passes through the logistic function $\sigma(x) = \frac{1}{1 + e^{-x}}$ to obtain output values between zero and one A threshold set to 0.5 would assign samples of outputs larger or equal 0.5 to the positive class and the rest to the negative class

If there are more than two classes $\sigma(x)$ itself would be a vector of size $n_{classes}$ Instead of passing through logistic function it passes through the softmax function which is written as

$$\text{softmax}(x) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

where x_i represents the i th element of the input to softmax which corresponds to class i and n is the number of classes The result is a vector containing the probabilities that sample x belong to each class The output is the class with the highest probability

In regression the output remains as $\sigma(x)$ therefore output activation function is just the identity function

MLP uses different loss functions depending on the problem type The loss function for classification is CrossEntropy which in binary case is given as

$$L = -\frac{1}{n} \sum_{i=1}^n [y_i \ln \sigma(x_i) + (1 - y_i) \ln (1 - \sigma(x_i))]$$

where n is

λ is an L2 regularization term aka penalty that penalizes complex models and $\lambda \geq 0$ is a nonnegative hyperparameter that controls the magnitude of the penalty

For regression MLP uses the Square Error loss function written as

$$L = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Starting from initial random weights multilayer perceptron MLP minimizes the loss function by repeatedly updating these weights After computing the loss a backward pass propagates it from the output layer to the previous layers providing each weight parameter with an update value meant to decrease the loss

In gradient descent the gradient $\nabla_{w_j} L$ of the loss with respect to the weights is computed and deducted from w_j More formally this is expressed as

$$w_j = w_j - \eta \nabla_{w_j} L$$

where j is the iteration step and η is the learning rate with a value larger than 0

The algorithm stops when it reaches a preset maximum number of iterations or when the improvement in loss is below a certain small number

scikitlearn user guide Release 0213

Tips on Practical Use

- Multilayer Perceptron is sensitive to feature scaling so it is highly recommended to scale your data For example scale each attribute on the input vector X to 0 1 or 1 1 or standardize it to have mean 0 and variance 1 Note that you must apply the same scaling to the test set for meaningful results You can use StandardScaler for standardization

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler
```

Dont cheat fit only on training data

```
scaler.fit(Xtrain)
```

```
Xtrain = scaler.transform(Xtrain)
```

apply same transformation to test data

```
Xtest = scaler.transform(Xtest)
```

An alternative and recommended approach is to use StandardScaler in a Pipeline

- Finding a reasonable regularization parameter λ is best done using GridSearchCV usually in the range 100 nparange1 7

- Empirically we observed that LBFGS converges faster and with better solutions on small datasets For relatively large datasets however Adam is very robust It usually converges quickly and gives pretty good performance Stochastic Gradient Descent with momentum or nesterov’s momentum on the other hand can perform better than those two algorithms if learning rate is correctly tuned

More control with warmstart

If you want more control over stopping criteria or learning rate in SGD or want to do additional monitoring using warmstart=True and max_iter=1 and iterating yourself can be helpful

```
X = [[0, 0, 1, 1],
     [0, 1, 0, 1]]
y = [0, 1]
clf = MLPClassifier(hidden_layer_sizes=(15, ), random_state=1, max_iter=1, warm
                    start=True)
for i in range(10):
    clf.fit(X, y)
    # additional monitoring / inspection
MLPClassifier
```

References

- “Learning representations by backpropagating errors” Rumelhart David E Geoffrey E Hinton and Ronald J Williams
- “Stochastic Gradient Descent” L Bottou Website 2010
- “Backpropagation” Andrew Ng Jiquan Ngiam Chuan Yu Foo Yifan Mai Caroline Suen Website 2011
- “Efficient BackProp” Y LeCun L Bottou G Orr K Müller In Neural Networks Tricks of the Trade 1998
- “Adam: A method for stochastic optimization” Kingma Diederik and Jimmy Ba arXiv preprint arXiv:1412.6980 2014

32 Unsupervised learning

321 Gaussian mixture models

sklearnmixture is a package which enables one to learn Gaussian Mixture Models diagonal spherical tied and full covariance matrices supported sample them and estimate them from data Facilities to help determine the appropriate number of components are also provided

Fig 33 Twocomponent Gaussian mixture model data points and equiprobability surfaces of the model

A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters One can think of mixture models as generalizing kmeans clustering to incorporate information about the covariance structure of the data as well as the centers of the latent Gaussians

Scikitlearn implements different classes to estimate Gaussian mixture models that correspond to different estimation strategies detailed below

Gaussian Mixture

TheGaussianMixture object implements the expectationmaximization EM algorithm for fitting mixtureof Gaussian models It can also draw confidence ellipsoids for multivariate models and compute the Bayesian Infor mation Criterion to assess the number of clusters in the data A GaussianMixturefit method is provided that learns a Gaussian Mixture Model from train data Given test data it can assign to each sample the Gaussian it mostly probably belong to using the GaussianMixturepredict method

TheGaussianMixture comes with different options to constrain the covariance of the difference classes estimated spherical diagonal tied or full covariance

Examples

- See GMM covariances for an example of using the Gaussian mixture as clustering on the iris dataset
- See Density Estimation for a Gaussian mixture for an example on plotting the density estimation

scikitlearn user guide Release 0213

Pros and cons of class GaussianMixture

Pros

Speed It is the fastest algorithm for learning mixture models

Agnostic As this algorithm maximizes only the likelihood it will not bias the means towards zero or bias the cluster sizes to have specific structures that might or might not apply

Cons

Singularities When one has insufficiently many points per mixture estimating the covariance matrices becomes difficult and the algorithm is known to diverge and find solutions with infinite likelihood unless one regularizes the covariances artificially

Number of components This algorithm will always use all the components it has access to needing heldout data or information theoretical criteria to decide how many components to use in the absence of external cues

Selecting the number of components in a classical Gaussian Mixture Model

The BIC criterion can be used to select the number of components in a Gaussian Mixture in an efficient way In theory it recovers the true number of components only in the asymptotic regime ie if much data is available and assuming

32 Unsupervised learning 337

scikitlearn user guide Release 0213

that the data was actually generated iid from a mixture of Gaussian distribution Note that using a Variational Bayesian Gaussian mixture avoids the specification of the number of components for a Gaussian mixture model

Examples

- See Gaussian Mixture Model Selection for an example of model selection performed with classical Gaussian mixture

Estimation algorithm Expectationmaximization

The main difficulty in learning Gaussian mixture models from unlabeled data is that it is one usually doesn't know which points came from which latent component if one has access to this information it gets very easy to fit a separate Gaussian distribution to each set of points Expectationmaximization is a wellfounded statistical algorithm to get around this problem by an iterative process First one assumes random components randomly centered on data points learned from kmeans or even just normally distributed around the origin and computes for each point a probability of being generated by each component of the model Then one tweaks the parameters to maximize the likelihood of the data given those assignments Repeating this process is guaranteed to always converge to a local optimum

Variational Bayesian Gaussian Mixture

TheBayesianGaussianMixture object implements a variant of the Gaussian mixture model with variational inference algorithms The API (Application Programming Interface) is similar as the one defined by GaussianMixture Estimation algorithm variational inference

Variational inference is an extension of expectationmaximization that maximizes a lower bound on model evidence including priors instead of data likelihood The principle behind variational methods is the same as expectation maximization that is both are iterative algorithms that alternate between finding the probabilities for each point to

be generated by each mixture and fitting the mixture to these assigned points but variational methods add regularization by integrating information from prior distributions This avoids the singularities often found in expectation maximization solutions but introduces some subtle biases to the model Inference is often notably slower but not usually as much so as to render usage unpractical

Due to its Bayesian nature the variational algorithm needs more hyper parameters than expectation maximization the most important of these being the concentration parameter `weightconcentrationprior` Specifying a low value for the concentration prior will make the model put most of the weight on few components set the remaining components weights very close to zero High values of the concentration prior will allow a larger number of components to be active in the mixture

The parameters implementation of the `BayesianGaussianMixture` class proposes two types of prior for the weights distribution a finite mixture model with Dirichlet distribution and an infinite mixture model with the Dirichlet Process In practice Dirichlet Process inference algorithm is approximated and uses a truncated distribution with a fixed maximum number of components called the Stickbreaking representation The number of components actually used almost always depends on the data

The next figure compares the results obtained for the different type of the weight concentration prior parameter `weightconcentrationpriortype` for different values of `weightconcentrationprior` Here we can see the value of the `weightconcentrationprior` parameter has a strong impact on the effective number of active components obtained We can also notice that large values for the concentration weight prior lead to more uniform weights when the type of prior is 'dirichletdistribution' while this is not necessarily the case for the 'dirichletprocess' type used by default

The examples below compare Gaussian mixture models with a fixed number of components to the variational Gaussian mixture models with a Dirichlet process prior. Here a classical Gaussian mixture is fitted with 5 components on a dataset composed of 2 clusters. We can see that the variational Gaussian mixture with a Dirichlet process prior is able to limit itself to only 2 components whereas the Gaussian mixture fits the data with a fixed number of components that has to be set a priori by the user. In this case the user has selected `ncomponents=5` which does not match the true generative distribution of this toy dataset. Note that with very little observations the variational Gaussian mixture models with a Dirichlet process prior can take a conservative stand and fit only one component. On the following figure we are fitting a dataset not well depicted by a Gaussian mixture. Adjusting the `weightconcentrationprior` parameter of the `BayesianGaussianMixture` controls the number of

340 Chapter 3 User Guide

scikitlearn user guide Release 0213

components used to fit this data We also present on the last two plots a random sampling generated from the two resulting mixtures

Examples

- See Gaussian Mixture Model Ellipsoids for an example on plotting the confidence ellipsoids for both GaussianMixture andBayesianGaussianMixture
 - Gaussian Mixture Model Sine Curve shows using GaussianMixture and BayesianGaussianMixture to fit a sine wave
 - See Concentration Prior Type Analysis of Variation Bayesian Gaussian Mixture for an example plotting the confidence ellipsoids for the BayesianGaussianMixture with dif
- 32 Unsupervised learning 341

scikitlearn user guide Release 0213

ferentweightconcentrationpriortype for different values of the parameter

weightconcentrationprior

Pros and cons of variational inference with BayesianGaussianMixture

Pros

Automatic selection whenweightconcentrationprior is small enough and ncomponents is larger than what is found necessary by the model the Variational Bayesian mixture model has a natural tendency to set some mixture weights values close to zero This makes it possible to let the model choose a suitable number of effective components automatically Only an upper bound of this number needs to be provided Note however that the “ideal” number of active components is very application specific and is typically illdefined in a data exploration setting Less sensitivity to the number of parameters unlike finite models which will almost always use all components as much as they can and hence will produce wildly different solutions for different numbers of components the variational inference with a Dirichlet process prior weightconcentrationpriortypedirichletprocess won’t change much with changes to the parameters leading to more stability and less tuning Regularization due to the incorporation of prior information variational solutions have less pathological special cases than expectationmaximization solutions

Cons

Speed the extra parametrization necessary for variational inference make inference slower although not by much Hyperparameters this algorithm needs an extra hyperparameter that might need experimental tuning via crossvalidation Bias there are many implicit biases in the inference algorithms and also in the Dirichlet process if used and whenever there is a mismatch between these biases and the data it might be possible to fit better models using a finite mixture

The Dirichlet Process

Here we describe variational inference algorithms on Dirichlet process mixture The Dirichlet process is a prior probability distribution on clusterings with an infinite unbounded number of partitions Variational techniques let us incorporate this prior structure on Gaussian mixture models at almost no penalty in inference time comparing with a finite Gaussian mixture model

An important question is how can the Dirichlet process use an infinite unbounded number of clusters and still be consistent While a full explanation doesn’t fit this manual one can think of its stick breaking process analogy to help understanding it The stick breaking process is a generative story for the Dirichlet process We start with a unitlength stick and in each step we break off a portion of the remaining stick Each time we associate the length of the piece of the stick to the proportion of points that falls into a group of the mixture At the end to represent the infinite mixture we associate the last remaining piece of the stick to the proportion of points that don’t fall into all the other groups The length of each piece is a random variable with probability proportional to the concentration parameter Smaller value of the concentration will divide the unitlength into larger pieces of the stick defining more concentrated distribution Larger concentration values will create smaller pieces of the stick increasing the number of components with non zero weights

scikitlearn user guide Release 0213

Variational inference techniques for the Dirichlet process still work with a finite approximation to this infinite mixture model but instead of having to specify a priori how many components one wants to use one just specifies the concentration parameter and an upper bound on the number of mixture components this upper bound assuming it is higher than the “true” number of components affects only algorithmic complexity not the actual number of components used

322 Manifold learning

Look for the bare necessities

The simple bare necessities

Forget about your worries and your strife

I mean the bare necessities

Old Mother Nature’s recipes

That bring the bare necessities of life

– Baloo’s song The Jungle Book

Manifold learning is an approach to nonlinear dimensionality reduction Algorithms for this task are based on the idea that the dimensionality of many data sets is only artificially high

Introduction

Highdimensional datasets can be very difficult to visualize While data in two or three dimensions can be plotted to show the inherent structure of the data equivalent highdimensional plots are much less intuitive To aid visualization of the structure of a dataset the dimension must be reduced in some way

The simplest way to accomplish this dimensionality reduction is by taking a random projection of the data Though this allows some degree of visualization of the data structure the randomness of the choice leaves much to be desired In a random projection it is likely that the more interesting structure within the data will be lost

32 Unsupervised learning 343

To address this concern a number of supervised and unsupervised linear dimensionality reduction frameworks have been designed such as Principal Component Analysis PCA (Principal Component Analysis) Independent Component Analysis Linear Discriminant Analysis and others These algorithms define specific rubrics to choose an “interesting” linear projection of the data These methods can be powerful but often miss important nonlinear structure in the data Manifold Learning can be thought of as an attempt to generalize linear frameworks like PCA (Principal Component Analysis) to be sensitive to non linear structure in data Though supervised variants exist the typical manifold learning problem is unsupervised it learns the highdimensional structure of the data from the data itself without the use of predetermined classifications

Examples

- See Manifold learning on handwritten digits Locally Linear Embedding Isomap for an example of dimensionality reduction on handwritten digits
- See Comparison of Manifold Learning methods for an example of dimensionality reduction on a toy “S curve” dataset

The manifold learning implementations available in scikitlearn are summarized below

Isomap

One of the earliest approaches to manifold learning is the Isomap algorithm short for Isometric Mapping Isomap can be viewed as an extension of Multidimensional Scaling MDS or Kernel PCA (Principal Component Analysis) Isomap seeks a lowerdimensional

344 Chapter 3 User Guide

scikitlearn user guide Release 0213

embedding which maintains geodesic distances between all points Isomap can be performed with the object Isomap Complexity

The Isomap algorithm comprises three stages

1Nearest neighbor search Isomap uses sklearnneighborsBallTree for efficient neighbor search

The cost is approximately $\log \log$ for nearest neighbors of points in dimensions

2Shortestpath graph search The most efficient known algorithms for this are Dijkstra’s Algorithm which is approximately $2 \log$ or the FloydWarshall algorithm which is 3 The algorithm can be selected by the user with the pathmethod keyword of Isomap If unspecified the code attempts to choose the best algorithm for the input data

3Partial eigenvalue decomposition The embedding is encoded in the eigenvectors corresponding to the largest eigenvalues of the \times isomap kernel For a dense solver the cost is approximately 2 This cost can often be improved using the ARPACK solver The eigensolver can be specified by the user with the pathmethod keyword of Isomap If unspecified the code attempts to choose the best algorithm for the input data

The overall complexity of Isomap is $\log \log 2 \log 2$

- number of training data points
- input dimension
- number of nearest neighbors
- output dimension

References

- “A global geometric framework for nonlinear dimensionality reduction” Tenenbaum JB De Silva V

Langford JC Science 290 5500

32 Unsupervised learning 345

scikitlearn user guide Release 0213

Locally Linear Embedding

Locally linear embedding LLE seeks a lowerdimensional projection of the data which preserves distances within local neighborhoods It can be thought of as a series of local Principal Component Analyses which are globally compared to find the best nonlinear embedding

Locally linear embedding can be performed with function `locallylinearemb` or its objectoriented counterpart `LocallyLinearEmbedding`

Complexity

The standard LLE algorithm comprises three stages

1Nearest Neighbors Search See discussion under Isomap above

2Weight Matrix Construction $\mathcal{O}(n^3)$ The construction of the LLE weight matrix involves the solution of $a \times n$ linear equation for each of the n local neighborhoods

3Partial Eigenvalue Decomposition See discussion under Isomap above

The overall complexity of standard LLE is $\mathcal{O}(n \log n \log n)$ $\mathcal{O}(n^3)$ $\mathcal{O}(n^2)$

- n number of training data points
- n input dimension
- n number of nearest neighbors
- n output dimension

References

- “Nonlinear dimensionality reduction by locally linear embedding” Roweis S Saul L Science 2902323 2000

Modified Locally Linear Embedding

One wellknown issue with LLE is the regularization problem When the number of neighbors is greater than the number of input dimensions the matrix defining each local neighborhood is rankdeficient To address this standard 346 Chapter 3 User Guide

LLE applies an arbitrary regularization parameter λ which is chosen relative to the trace of the local weight matrix. Though it can be shown formally that as $\lambda \rightarrow 0$ the solution converges to the desired embedding there is no guarantee that the optimal solution will be found for $\lambda = 0$. This problem manifests itself in embeddings which distort the underlying geometry of the manifold.

One method to address the regularization problem is to use multiple weight vectors in each neighborhood.

This is the essence of modified locally linear embedding (MLLE). MLLE can be performed with function

`locallylinearembd` or its object-oriented counterpart `LocallyLinearEmbedding` with the key

`wordmethod` modified. It requires `nneighbors` `ncomponents`.

Complexity

The MLLE algorithm comprises three stages:

1. Nearest Neighbors Search: Same as standard LLE.

2. Weight Matrix Construction: Approximately $O(n^3 + k^2n)$. The first term is exactly equivalent to that of standard LLE. The second term has to do with constructing the weight matrix from multiple weights. In practice the added cost of constructing the MLLE weight matrix is relatively small compared to the cost of steps 1 and 3.

3. Partial Eigenvalue Decomposition: Same as standard LLE.

The overall complexity of MLLE is $O(n \log n + k^2 \log n + k^3 + k^2n)$.

- n : number of training data points
- k : input dimension
- k : number of nearest neighbors
- k : output dimension

References

- “MLLE: Modified Locally Linear Embedding Using Multiple Weights” Zhang Z. Wang J.

scikitlearn user guide Release 0213

Hessian Eigenmapping

Hessian Eigenmapping also known as Hessianbased LLE HLLLE is another method of solving the regularization problem of LLE It revolves around a hessianbased quadratic form at each neighborhood which is used to recover the locally linear structure Though other implementations note its poor scaling with data size sklearn implements some algorithmic improvements which make its cost comparable to that of other LLE variants for small output dimension HLLLE can be performed with function locallylinearembembedding or its objectoriented counterpart LocallyLinearEmbedding with the keyword method hessian It requires nneighbors

ncomponents ncomponents 3 2

Complexity

The HLLLE algorithm comprises three stages

1Nearest Neighbors Search Same as standard LLE

2Weight Matrix Construction Approximately $O(n^3 \log n)$ The first term reflects a similar cost to that of standard LLE The second term comes from a QR decomposition of the local hessian estimator

3Partial Eigenvalue Decomposition Same as standard LLE

The overall complexity of standard HLLLE is $O(n \log n \log n)$

- n number of training data points
- d input dimension
- k number of nearest neighbors
- d' output dimension

References

- “Hessian Eigenmaps Locally linear embedding techniques for highdimensional data” Donoho D Grimes C Proc Natl Acad Sci USA 1005591 2003

Spectral Embedding

Spectral Embedding is an approach to calculating a nonlinear embedding Scikitlearn implements Laplacian Eigen maps which finds a low dimensional representation of the data using a spectral decomposition of the graph Laplacian The graph generated can be considered as a discrete approximation of the low dimensional manifold in the high dimensional space Minimization of a cost function based on the graph ensures that points close to each other on the manifold are mapped close to each other in the low dimensional space preserving local distances Spectral embedding can be performed with the function `spectralembedding` or its objectoriented counterpart `SpectralEmbedding`

Complexity

The Spectral Embedding Laplacian Eigenmaps algorithm comprises three stages

1Weighted Graph Construction Transform the raw input data into graph representation using affinity adjacency matrix representation

2Graph Laplacian Construction unnormalized Graph Laplacian is constructed as $L = D - A$ for and normalized one as $L_{\text{norm}} = D^{-1/2} L D^{-1/2}$

2 D - A - 1

2

3Partial Eigenvalue Decomposition Eigenvalue decomposition is done on graph Laplacian

The overall complexity of spectral embedding is $O(n \log n \log n + n^3 + n^2)$

- n number of training data points
- n input dimension
- n number of nearest neighbors
- n output dimension

References

- “Laplacian Eigenmaps for Dimensionality Reduction and Data Representation” M Belkin P Niyogi Neural Computation June 2003 15 613731396

Local Tangent Space Alignment

Though not technically a variant of LLE Local tangent space alignment LTSA is algorithmically similar enough to LLE that it can be put in this category Rather than focusing on preserving neighborhood distances as in LLE LTSA seeks to characterize the local geometry at each neighborhood via its tangent space and performs a global optimization to align these local tangent spaces to learn the embedding LTSA can be performed with function `locallylinearembembedding` or its objectoriented counterpart `LocallyLinearEmbedding` with the key word `method Itsa`

Complexity

The LTSA algorithm comprises three stages

1Nearest Neighbors Search Same as standard LLE

2Weight Matrix Construction Approximately $O(n^3 + n^2)$ The first term reflects a similar cost to that of standard LLE

3Partial Eigenvalue Decomposition Same as standard LLE

32 Unsupervised learning 349

scikitlearn user guide Release 0213

The overall complexity of standard LTSA is $O(n \log n \log n)$ where n is the number of training data points

- n number of training data points
- d input dimension
- k number of nearest neighbors
- m output dimension

References

- “Principal manifolds and nonlinear dimensionality reduction via tangent space alignment” Zhang Z Zha

H Journal of Shanghai Univ 8406 2004

Multidimensional Scaling MDS

Multidimensional scaling MDS seeks a lowdimensional representation of the data in which the distances respect well the distances in the original highdimensional space

In general is a technique used for analyzing similarity or dissimilarity data MDS attempts to model similarity or dissimilarity data as distances in a geometric spaces The data can be ratings of similarity between objects interaction frequencies of molecules or trade indices between countries

There exists two types of MDS algorithm metric and non metric In the scikitlearn the class MDS implements both In Metric MDS the input similarity matrix arises from a metric and thus respects the triangular inequality the distances between output two points are then set to be as close as possible to the similarity or dissimilarity data In the nonmetric version the algorithms will try to preserve the order of the distances and hence seek for a monotonic relationship between the distances in the embedded space and the similaritiesdissimilarities

Let S be the similarity matrix and X the coordinates of the n input points Disparities Δ are transformation of the similarities chosen in some optimal ways The objective called the stress is then defined by $\sum_{i,j} (\Delta_{ij} - d_{ij})^2$

Metric MDS

The simplest metric MDS model called absolute MDS disparities are defined by $\Delta_{ij} = d_{ij}$ With absolute MDS the value Δ_{ij} should then correspond exactly to the distance between point i and j in the embedding point

scikitlearn user guide Release 0213

Most commonly disparities are set to $\frac{1}{2}(d_{ij}^2 - d_{ij}^*)^2$

Nonmetric MDS

Non metric MDS focuses on the ordination of the data If d_{ij} is the distance between x_i and x_j then the embedding should enforce $d_{ij} \leq d_{ij}^*$

A simple algorithm to enforce that is to use a monotonic regression of d_{ij}^* on d_{ij} yielding disparities d_{ij}^* in the same order as d_{ij}

A trivial solution to this problem is to set all the points on the origin In order to avoid that the disparities d_{ij}^* are normalized

References

- “Modern Multidimensional Scaling Theory and Applications” Borg I Groenen P Springer Series in Statistics 1997
 - “Nonmetric multidimensional scaling a numerical method” Kruskal J Psychometrika 29 1964
- 32 Unsupervised learning 351

scikitlearn user guide Release 0213

- “Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis” Kruskal J Psychometrika 29 1964

tdistributed Stochastic Neighbor Embedding tSNE

tSNE TSNE converts affinities of data points to probabilities The affinities in the original space are represented by Gaussian joint probabilities and the affinities in the embedded space are represented by Student’s tdistributions This allows tSNE to be particularly sensitive to local structure and has a few other advantages over existing techniques

- Revealing the structure at many scales on a single map
- Revealing data that lie in multiple different manifolds or clusters
- Reducing the tendency to crowd points together at the center

While Isomap LLE and variants are best suited to unfold a single continuous low dimensional manifold tSNE will focus on the local structure of the data and will tend to extract clustered local groups of samples as highlighted on the Scurve example This ability to group samples based on the local structure might be beneficial to visually disentangle a dataset that comprises several manifolds at once as is the case in the digits dataset

The KullbackLeibler KL divergence of the joint probabilities in the original space and the embedded space will be minimized by gradient descent Note that the KL divergence is not convex ie multiple restarts with different initializations will end up in local minima of the KL divergence Hence it is sometimes useful to try different seeds and select the embedding with the lowest KL divergence

The disadvantages to using tSNE are roughly

- tSNE is computationally expensive and can take several hours on millionsample datasets where PCA (Principal Component Analysis) will finish in seconds or minutes
- The BarnesHut tSNE method is limited to two or three dimensional embeddings
- The algorithm is stochastic and multiple restarts with different seeds can yield different embeddings However it is perfectly legitimate to pick the embedding with the least error
- Global structure is not explicitly preserved This problem is mitigated by initializing points with PCA (Principal Component Analysis) using initpca

Optimizing tSNE

The main purpose of tSNE is visualization of highdimensional data Hence it works best when the data will be embedded on two or three dimensions

Optimizing the KL divergence can be a little bit tricky sometimes There are five parameters that control the optimization of tSNE and therefore possibly the quality of the resulting embedding

- perplexity
- early exaggeration factor
- learning rate
- maximum number of iterations
- angle not used in the exact method

The perplexity is defined as $2^{\frac{1}{\log_2 \frac{1}{\sum_{j=1}^k p_j}}}$ where $\frac{1}{\sum_{j=1}^k p_j}$ is the Shannon entropy of the conditional probability distribution The perplexity of a k -sided die is k so that $\frac{1}{\sum_{j=1}^k p_j}$ is effectively the number of nearest neighbors tSNE considers when generating the conditional probabilities Larger perplexities lead to more nearest neighbors and less sensitive to small structure Conversely a lower perplexity considers a smaller number of neighbors and thus ignores more global information in favour of the local neighborhood As dataset sizes get larger more points will be required to get a reasonable sample of the local neighborhood and hence larger perplexities may be required Similarly noisier datasets will require larger perplexity values to encompass enough local neighbors to see beyond the background noise The maximum number of iterations is usually high enough and does not need any tuning The optimization consists of two phases the early exaggeration phase and the final optimization During early exaggeration the joint probabilities in the original space will be artificially increased by multiplication with a given factor Larger factors result in larger gaps between natural clusters in the data If the factor is too high the KL divergence could increase during this phase Usually it does not have to be tuned A critical parameter is the learning rate If it is too low gradient descent will get stuck in a bad local minimum If it is too high the KL divergence will increase during optimization More tips can be found in Laurens van der Maaten’s FAQ see references The last parameter angle is a tradeoff between performance and accuracy Larger angles imply that we can approximate larger regions by a single point leading to better speed but less accurate results

“How to Use tSNE Effectively” provides a good discussion of the effects of the various parameters as well as interactive plots to explore the effects of different parameters

BarnesHut tSNE

The BarnesHut tSNE that has been implemented here is usually much slower than other manifold learning algorithms The optimization is quite difficult and the computation of the gradient is $O(n^2 \log n)$ where n is the number of output dimensions and n is the number of samples The BarnesHut method improves on the exact method where tSNE complexity is $O(n^3)$ but has several other notable differences

- The BarnesHut implementation only works when the target dimensionality is 3 or less The 2D case is typical when building visualizations
- BarnesHut only works with dense input data Sparse data matrices can only be embedded with the exact method or can be approximated by a dense low rank projection for instance using `sklearn.decomposition`

TruncatedSVD

- BarnesHut is an approximation of the exact method The approximation is parameterized with the angle parameter therefore the angle parameter is unused when method “exact”
- BarnesHut is significantly more scalable BarnesHut can be used to embed hundred of thousands of data points while the exact method can handle thousands of samples before becoming computationally intractable

scikitlearn user guide Release 0213

For visualization purpose which is the main use case of tSNE using the BarnesHut method is strongly recommended. The exact tSNE method is useful for checking the theoretically properties of the embedding possibly in higher dimensional space but limit to small datasets due to computational constraints. Also note that the digits labels roughly match the natural grouping found by tSNE while the linear 2D projection of the PCA (Principal Component Analysis) model yields a representation where label regions largely overlap. This is a strong clue that this data can be well separated by non linear methods that focus on the local structure eg an SVM (Support Vector Machine) with a Gaussian RBF (Radial Basis Function). However failing to visualize well separated homogeneously labeled groups with tSNE in 2D does not necessarily imply that the data cannot be correctly classified by a supervised model. It might be the case that 2 dimensions are not low enough to accurately represents the internal structure of the data.

References

- “Visualizing HighDimensional Data Using tSNE” van der Maaten LJP Hinton G Journal of Machine Learning Research 2008
- “tDistributed Stochastic Neighbor Embedding” van der Maaten LJP
- “Accelerating tSNE using TreeBased Algorithms” LJP van der Maaten Journal of Machine Learning Research 15Oct32213245 2014

Tips on practical use

- Make sure the same scale is used over all features. Because manifold learning methods are based on a nearest neighbor search the algorithm may perform poorly otherwise. See StandardScaler for convenient ways of scaling heterogeneous data.
- The reconstruction error computed by each routine can be used to choose the optimal output dimension. For a d -dimensional manifold embedded in a D -dimensional parameter space the reconstruction error will decrease as n components is increased until n components $\approx d$.
- Note that noisy data can “shortcircuit” the manifold in essence acting as a bridge between parts of the manifold that would otherwise be well separated. Manifold learning on noisy and/or incomplete data is an active area of research.
- Certain input configurations can lead to singular weight matrices for example when more than two points in the dataset are identical or when the data is split into disjointed groups. In this case solver arpack will fail to find the null space. The easiest way to address this is to use solver dense which will work on a singular matrix though it may be very slow depending on the number of input points. Alternatively one can attempt to understand the source of the singularity if it is due to disjoint sets increasing `n_neighbors` may help. If it is due to identical points in the dataset removing these points may help.

See also

Totally Random Trees Embedding can also be useful to derive nonlinear representations of feature space also it does not perform dimensionality reduction.

323 Clustering

Clustering of unlabeled data can be performed with the module `sklearn.cluster`.

Each clustering algorithm comes in two variants: a class that implements the `fit` method to learn the clusters on train data and a function that given train data returns an array of integer labels corresponding to the different clusters. For the class the labels over the training data can be found in the `labels` attribute.

354 Chapter 3 User Guide

Input data

One important thing to note is that the algorithms implemented in this module can take different kinds of matrix as input. All the methods accept standard data matrices of shape `nsamples nfeatures`. These can be obtained from the classes in the `sklearnfeatureextraction` module. For `AffinityPropagation`, `SpectralClustering` and `DBSCAN` one can also input similarity matrices of shape `nsamples nsamples`. These can be obtained from the functions in the `sklearnmetricspairwise` module.

Overview of clustering methods

Fig 34 A comparison of the clustering algorithms in scikitlearn

scikitlearn user guide Release 0.21.3

points from μ although they live in the same space

The Kmeans algorithm aims to choose centroids that minimise the inertia or within cluster sum of squares criterion

$$\sum$$

$$\mu \in \mathbb{R}^d$$

$$\mu \in \mathbb{R}^d - \mathbb{R}^d$$

Inertia can be recognized as a measure of how internally coherent clusters are. It suffers from various drawbacks

- Inertia makes the assumption that clusters are convex and isotropic which is not always the case. It responds poorly to elongated clusters or manifolds with irregular shapes.
- Inertia is not a normalized metric; we just know that lower values are better and zero is optimal. But in very high-dimensional spaces, Euclidean distances tend to become inflated; this is an instance of the so-called “curse of dimensionality”. Running a dimensionality reduction algorithm such as Principal Component Analysis (PCA) prior to kmeans clustering can alleviate this problem and speed up the computations.

K-means is often referred to as Lloyd’s algorithm. In basic terms, the algorithm has three steps. The first step chooses the initial centroids with the most basic method being to choose k samples from the dataset. After initialization, 32 Unsupervised learning 357

scikitlearn user guide Release 0213

Kmeans consists of looping between the two other steps The first step assigns each sample to its nearest centroid The second step creates new centroids by taking the mean value of all of the samples assigned to each previous centroid The difference between the old and the new centroids are computed and the algorithm repeats these last two steps until this value is less than a threshold In other words it repeats until the centroids do not move significantly Kmeans is equivalent to the expectationmaximization algorithm with a small allequal diagonal covariance matrix

The algorithm can also be understood through the concept of V oronoi diagrams First the V oronoi diagram of the points is calculated using the current centroids Each segment in the V oronoi diagram becomes a separate cluster Secondly the centroids are updated to the mean of each segment The algorithm then repeats this until a stopping criterion is fulfilled Usually the algorithm stops when the relative decrease in the objective function between iterations is less than the given tolerance value This is not the case in this implementation iteration stops when centroids move less than the tolerance

Given enough time Kmeans will always converge however this may be to a local minimum This is highly depen dent on the initialization of the centroids As a result the computation is often done several times with different initializations of the centroids One method to help address this issue is the kmeans initialization scheme which has been implemented in scikitlearn use the initkmeans parameter This initializes the centroids to be generally distant from each other leading to provably better results than random initialization as shown in the reference

The algorithm supports sample weights which can be given by a parameter sampleweight This allows to assign more weight to some samples when computing cluster centers and values of inertia For example assigning a weight of 2 to a sample is equivalent to adding a duplicate of that sample to the dataset

A parameter can be given to allow Kmeans to be run in parallel called njobs Giving this parameter a positive value uses that many processors default 1 A value of 1 uses all available processors with 2 using one less and so on Parallelization generally speeds up computation at the cost of memory in this case multiple copies of centroids need to be stored one for each job

Warning The parallel version of KMeans is broken on OS X when numpy uses theAccelerate Framework

This is expected behavior Accelerate can be called after a fork but you need to execv the subprocess with the Python binary which multiprocessing does not do under posix

Kmeans can be used for vector quantization This is achieved using the transform method of a trained model of KMeans

Examples

- Demonstration of kmeans assumptions Demonstrating when kmeans performs intuitively and when it does not

- A demo of KMeans clustering on the handwritten digits data Clustering handwritten digits

References

- “kmeans The advantages of careful seeding” Arthur David and Sergei Vassilvitskii Proceedings of the eighteenth annual ACMSIAM symposium on Discrete algorithms Society for Industrial and Applied Mathematics 2007

Mini Batch KMeans

TheMiniBatchKMeans is a variant of the KMeans algorithm which uses minibatches to reduce the computation time while still attempting to optimise the same objective function Minibatches are subsets of the input data randomly sampled in each training iteration These minibatches drastically reduce the amount of computation required to converge to a local solution In contrast to other algorithms that reduce the convergence time of kmeans minibatch kmeans produces results that are generally only slightly worse than the standard algorithm

The algorithm iterates between two major steps similar to vanilla kmeans In the first step n samples are drawn randomly from the dataset to form a minibatch These are then assigned to the nearest centroid In the second step the centroids are updated In contrast to kmeans this is done on a per sample basis For each sample in the minibatch the assigned centroid is updated by taking the streaming average of the sample and all previous samples assigned to that centroid This has the effect of decreasing the rate of change for a centroid over time These steps are performed until convergence or a predetermined number of iterations is reached

MiniBatchKMeans converges faster than KMeans but the quality of the results is reduced In practice this difference in quality can be quite small as shown in the example and cited reference

Examples

- Comparison of the KMeans and MiniBatchKMeans clustering algorithms Comparison of KMeans and MiniBatchKMeans
- Clustering text documents using kmeans Document clustering using sparse MiniBatchKMeans
- Online learning of a dictionary of parts of faces

References

- “Web Scale KMeans clustering” D Sculley Proceedings of the 19th international conference on World wide web2010

Affinity Propagation

AffinityPropagation creates clusters by sending messages between pairs of samples until convergence A dataset is then described using a small number of exemplars which are identified as those most representative of other samples The messages sent between pairs represent the suitability for one sample to be the exemplar of the other which is updated in response to the values from other pairs This updating happens iteratively until convergence at which point the final exemplars are chosen and hence the final clustering is given Affinity Propagation can be interesting as it chooses the number of clusters based on the data provided For this purpose the two important parameters are the preference which controls how many exemplars are used and the damping factor which damps the responsibility and availability messages to avoid numerical oscillations when updating these messages

The main drawback of Affinity Propagation is its complexity The algorithm has a time complexity of the order $O(n^2 \cdot i)$ where n is the number of samples and i is the number of iterations until convergence Further the memory complexity is of the order $O(n^2)$ if a dense similarity matrix is used but reducible if a sparse similarity matrix is used This makes Affinity Propagation most appropriate for small to medium sized datasets

Examples

- Demo of affinity propagation clustering algorithm Affinity Propagation on a synthetic 2D datasets with 3 classes
- Visualizing the stock market structure Affinity Propagation on Financial time series to find groups of companies

Algorithm description The messages sent between points belong to one of two categories The first is the responsibility r_{ij} which is the accumulated evidence that sample j should be the exemplar for sample i The second is the availability a_{ij} which is the accumulated evidence that sample i should choose sample j to be its exemplar and considers the values for all other samples that j should be an exemplar In this way exemplars are chosen by samples if they are 1 similar enough to many samples and 2 chosen by many samples to be representative of themselves

More formally the responsibility of a sample i to be the exemplar of sample j is given by

$$r_{ji} \leftarrow \frac{1 - \max_{k \neq i} s_{jk}}{\sum_{k=1}^K (1 - \max_{k \neq i} s_{jk})}$$

Where s_{ji} is the similarity between samples i and j . The availability of sample i to be the exemplar of sample j is given by

$$a_{ji} \leftarrow \frac{1}{\sum_{k=1}^K r_{jk}}$$

$$r'_{ji} \in [0, 1]$$

To begin with all values for r and a are set to zero and the calculation of each iterates until convergence As discussed above in order to avoid numerical oscillations when updating the messages the damping factor α is introduced to iteration process

$$r'_{ji} \leftarrow \alpha \cdot r_{ji} + (1 - \alpha) \cdot a_{ji}$$

$$a'_{ji} \leftarrow \alpha \cdot a_{ji} + (1 - \alpha) \cdot r_{ji}$$

where t indicates the iteration times

Mean Shift

MeanShift clustering aims to discover blobs in a smooth density of samples It is a centroid based algorithm which works by updating candidates for centroids to be the mean of the points within a given region These candidates are then filtered in a postprocessing stage to eliminate nearduplicates to form the final set of centroids

Given a candidate centroid μ for iteration t the candidate is updated according to the following equation

$$\mu^{t+1}$$

$$= \frac{1}{n} \sum_{x \in N(\mu)}$$

$$\mu$$

Where $N(\mu)$ is the neighborhood of samples within a given distance around μ and μ is the mean shift vector that is computed for each centroid that points towards a region of the maximum increase in the density of points This is computed using the following equation effectively updating a centroid to be the mean of the samples within its neighborhood

$$\mu^{t+1} = \mu^t + \frac{1}{n} \sum_{x \in N(\mu^t)} (x - \mu^t)$$

$$\mu^t \in \mathbb{R}^d$$

$$\mu^t \in \mathbb{R}^d$$

The algorithm automatically sets the number of clusters instead of relying on a parameter bandwidth which dictates the size of the region to search through This parameter can be set manually but can be estimated using the provided estimatebandwidth function which is called if the bandwidth is not set

The algorithm is not highly scalable as it requires multiple nearest neighbor searches during the execution of the algorithm The algorithm is guaranteed to converge however the algorithm will stop iterating when the change in centroids is small

Labelling a new sample is performed by finding the nearest centroid for a given sample

Examples

- A demo of the meanshift clustering algorithm Mean Shift clustering on a synthetic 2D datasets with 3 classes

scikitlearn user guide Release 0213

References

- “Mean shift A robust approach toward feature space analysis” D Comaniciu and P Meer IEEE Transactions on Pattern Analysis and Machine Intelligence 2002

Spectral clustering

SpectralClustering does a lowdimension embedding of the affinity matrix between samples followed by a KMeans in the low dimensional space It is especially efficient if the affinity matrix is sparse and the pyamg module is installed SpectralClustering requires the number of clusters to be specified It works well for a small number of clusters but is not advised when using many clusters

For two clusters it solves a convex relaxation of the normalised cuts problem on the similarity graph cutting the graph in two so that the weight of the edges cut is small compared to the weights of the edges inside each cluster This criteria is especially interesting when working on images graph vertices are pixels and edges of the similarity graph are a function of the gradient of the image

Warning Transforming distance to wellbehaved similarities

Note that if the values of your similarity matrix are not well distributed eg with negative values or with a distance

scikitlearn user guide Release 0213

matrix rather than a similarity the spectral problem will be singular and the problem not solvable In which case it is advised to apply a transformation to the entries of the matrix For instance in the case of a signed distance matrix is common to apply a heat kernel
similarity npexpbeta distance distancestd
See the examples for such an application

Examples

- Spectral clustering for image segmentation Segmenting objects from a noisy background using spectral clustering

- Segmenting the picture of greek coins in regions Spectral clustering to split the image of coins in regions

Different label assignment strategies

Different label assignment strategies can be used corresponding to the assignlabels parameter of SpectralClustering Thekmeans strategy can match finer details of the data but it can be more unstable In particular unless you control the randomstate it may not be reproducible from runtorun as it depends on a random initialization On the other hand the discretize strategy is 100 reproducible but it tends to create parcels of fairly even and geometrical shape
assignlabelskmeans assignlabelsdiscretize

Spectral Clustering Graphs

Spectral Clustering can also be used to cluster graphs by their spectral embeddings In this case the affinity matrix is the adjacency matrix of the graph and SpectralClustering is initialized with affinityprecomputed

32 Unsupervised learning 363

```
scikitlearn user guide Release 0213
from sklearncluster import SpectralClustering
sc SpectralClustering3 affinityprecomputed ninit100
assignlabelsdiscretize
scfitpredictadjacencymatrix
```

References

- “A Tutorial on Spectral Clustering” Ulrike von Luxburg 2007
- “Normalized cuts and image segmentation” Jianbo Shi Jitendra Malik 2000
- “A Random Walks View of Spectral Segmentation” Marina Meila Jianbo Shi 2001
- “On Spectral Clustering Analysis and an algorithm” Andrew Y Ng Michael I Jordan Yair Weiss 2001

Hierarchical clustering

Hierarchical clustering is a general family of clustering algorithms that build nested clusters by merging or splitting them successively This hierarchy of clusters is represented as a tree or dendrogram The root of the tree is the unique cluster that gathers all the samples the leaves being the clusters with only one sample See the Wikipedia page for more details

TheAgglomerativeClustering object performs a hierarchical clustering using a bottom up approach each observation starts in its own cluster and clusters are successively merged together The linkage criteria determines the metric used for the merge strategy

- Ward minimizes the sum of squared differences within all clusters It is a varianceminimizing approach and in this sense is similar to the kmeans objective function but tackled with an agglomerative hierarchical approach
- Maximum orcomplete linkage minimizes the maximum distance between observations of pairs of clusters
- Average linkage minimizes the average of the distances between all observations of pairs of clusters
- Single linkage minimizes the distance between the closest observations of pairs of clusters

AgglomerativeClustering can also scale to large number of samples when it is used jointly with a connectivity matrix but is computationally expensive when no connectivity constraints are added between samples it considers at each step all the possible merges

FeatureAgglomeration

TheFeatureAgglomeration uses agglomerative clustering to group together features that look very similar thus decreasing the number of features It is a dimensionality reduction tool see Unsupervised dimensionality reduction

Different linkage type Ward complete average and single linkage

AgglomerativeClustering supports Ward single average and complete linkage strategies

Ag

glomerative cluster has a “rich get richer” behavior that leads to uneven cluster sizes In this regard single linkage is the worst strategy and Ward gives the most regular sizes However the affinity or distance used in clustering cannot be varied with Ward thus for non Euclidean metrics average linkage is a good alternative Single linkage while not robust to noisy data can be computed very efficiently and can therefore be useful to provide hierarchical clustering of larger datasets Single linkage can also perform well on nonglobular data

Examples

- Various Agglomerative Clustering on a 2D embedding of digits exploration of the different linkage strategies in a real dataset

Adding connectivity constraints

An interesting aspect of AgglomerativeClustering is that connectivity constraints can be added to this algorithm only adjacent clusters can be merged together through a connectivity matrix that defines for each sample the neighboring samples following a given structure of the data For instance in the swissroll example below the connectivity constraints forbid the merging of points that are not adjacent on the swiss roll and thus avoid forming clusters that extend across overlapping folds of the roll
These constraint are useful to impose a certain local structure but they also make the algorithm faster especially when the number of the samples is high

The connectivity constraints are imposed via an connectivity matrix a scipy sparse matrix that has elements only at the intersection of a row and a column with indices of the dataset that should be connected This matrix can be constructed from apriori information for instance you may wish to cluster web pages by only merging pages with a link pointing from one to another It can also be learned from the data for instance using sklearn.neighbors.kneighborsgraph to restrict merging to nearest neighbors as in this example or using sklearn.feature_extraction.image.grid_to_graph to enable only merging of neighboring pixels on an image as in the coin example

Examples

- A demo of structured Ward hierarchical clustering on an image of coins Ward clustering to split the image of coins in regions
- Hierarchical clustering structured vs unstructured ward Example of Ward algorithm on a swissroll comparison of structured approaches versus unstructured approaches
- Feature agglomeration vs univariate selection Example of dimensionality reduction with feature agglomeration based on Ward hierarchical clustering
- Agglomerative clustering with and without structure

Warning Connectivity constraints with single average and complete linkage
Connectivity constraints and single complete or average linkage can enhance the ‘rich getting richer’ aspect of agglomerative clustering particularly so if they are built with sklearn.neighbors.kneighborsgraph
In the limit of a small number of clusters they tend to give a few macroscopically occupied clusters and almost empty ones see the discussion in Agglomerative clustering with and without structure Single linkage is the most brittle linkage option with regard to this issue

Varying the metric

Single average and complete linkage can be used with a variety of distances or affinities in particular Euclidean distance l2 Manhattan distance or Cityblock or l1 cosine distance or any precomputed affinity matrix

- l1distance is often good for sparse features or sparse noise ie many of the features are zero as in text mining using occurrences of rare words

- cosine distance is interesting because it is invariant to global scalings of the signal

The guidelines for choosing a metric is to use one that maximizes the distance between samples in different classes and minimizes that within each class

Examples

- Agglomerative clustering with different metrics

DBSCAN

TheDBSCAN algorithm views clusters as areas of high density separated by areas of low density Due to this rather generic view clusters found by DBSCAN can be any shape as opposed to kmeans which assumes that clusters are convex shaped The central component to the DBSCAN is the concept of core samples which are samples that are in areas of high density A cluster is therefore a set of core samples each close to each other measured by some distance measure and a set of noncore samples that are close to a core sample but are not themselves core samples There are two parameters to the algorithm minsamples andeps which define formally what we mean when we say dense Higherminsamples or lowereps indicate higher density necessary to form a cluster

More formally we define a core sample as being a sample in the dataset such that there exist minsamples other samples within a distance of eps which are defined as neighbors of the core sample This tells us that the core sample is in a dense area of the vector space A cluster is a set of core samples that can be built by recursively taking a core sample finding all of its neighbors that are core samples finding all of their neighbors that are core samples and so on A cluster also has a set of noncore samples which are samples that are neighbors of a core sample in the cluster but are not themselves core samples Intuitively these samples are on the fringes of a cluster

scikitlearn user guide Release 0213

Any core sample is part of a cluster by definition Any sample that is not a core sample and is at least eps in distance from any core sample is considered an outlier by the algorithm

While the parameter minsamples primarily controls how tolerant the algorithm is towards noise on noisy and large data sets it may be desiable to increase this parameter the parameter eps iscrucial to choose appropriately for the data set and distance function and usually cannot be left at the default value It controls the local neighborhood of the points When chosen too small most data will not be clustered at all and labeled as 1for “noise” When chosen too large it causes close clusters to be merged into one cluster and eventually the entire data set to be returned as a single cluster Some heuristics for choosing this parameter have been discussed in literature for example based on a knee in the nearest neighbor distances plot as discussed in the references below

In the figure below the color indicates cluster membership with large circles indicating core samples found by the algorithm Smaller circles are noncore samples that are still part of a cluster Moreover the outliers are indicated by black points below

Examples

- Demo of DBSCAN clustering algorithm

Implementation

The DBSCAN algorithm is deterministic always generating the same clusters when given the same data in the same order However the results can differ when data is provided in a different order First even though the core samples will always be assigned to the same clusters the labels of those clusters will depend on the order in which those samples are encountered in the data Second and more importantly the clusters to which noncore samples are assigned can differ depending on the data order This would happen when a noncore sample has a distance lower thaneps to two core samples in different clusters By the triangular inequality those two core samples must be more distant than eps from each other or they would be in the same cluster The noncore sample is assigned to whichever cluster is generated first in a pass through the data and so the results will depend on the data ordering The current implementation uses ball trees and kdtrees to determine the neighborhood of points which avoids calculating the full distance matrix as was done in scikitlearn versions before 014 The possibility to use custom metrics is retained for details see NearestNeighbors

Memory consumption for large sample sizes

368 Chapter 3 User Guide

scikitlearn user guide Release 0213

This implementation is by default not memory efficient because it constructs a full pairwise similarity matrix in the case where kdtrees or balltrees cannot be used eg with sparse matrices This matrix will consume n^2 floats

A couple of mechanisms for getting around this are

- Use OPTICS clustering in conjunction with the extractdbscan method OPTICS clustering also calculates the full pairwise matrix but only keeps one row in memory at a time memory complexity n
- A sparse radius neighborhood graph where missing entries are presumed to be out of eps can be precomputed in a memoryefficient way and dbscan can be run over this with metricprecomputed See `sklearn.neighbors.NearestNeighbors.radius_neighbors_graph`
- The dataset can be compressed either by removing exact duplicates if these occur in your data or by using BIRCH Then you only have a relatively small number of representatives for a large number of points You can then provide a sampleweight when fitting DBSCAN

References

- “A DensityBased Algorithm for Discovering Clusters in Large Spatial Databases with Noise” Ester M H P Kriegel J Sander and X Xu In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining Portland OR AAAI Press pp 226–231 1996
- “DBSCAN revisited revisited why and how you should still use DBSCAN Schubert E Sander J Ester M Kriegel H P Xu X 2017 In ACM Transactions on Database Systems TODS 423 19

OPTICS

TheOPTICS algorithm shares many similarities with the DBSCAN algorithm and can be considered a generalization of DBSCAN that relaxes the eps requirement from a single value to a value range The key difference between DBSCAN and OPTICS is that the OPTICS algorithm builds a reachability graph which assigns each sample both areachability distance and a spot within the cluster ordering attribute these two attributes are assigned when the model is fitted and are used to determine cluster membership If OPTICS is run with the default value of `inf` for `maxeps` then DBSCAN style cluster extraction can be performed repeatedly in linear time for any given eps value using the `cluster_optics_dbscan` method Setting `maxeps` to a lower value will result in shorter run times and can be thought of as the maximum neighborhood radius from each point to find other potential reachable points

Thereachability distances generated by OPTICS allow for variable density extraction of clusters within a single data set As shown in the above plot combining reachability distances and data set ordering produces a reachability plot where point density is represented on the Yaxis and points are ordered such that nearby points are adjacent 'Cutting' the reachability plot at a single value produces DBSCAN like results all points above the 'cut' are classified as noise and each time that there is a break when reading from left to right signifies a new cluster The default cluster extraction with OPTICS looks at the steep slopes within the graph to find clusters and the user can define what counts as a steep slope using the parameter ξ There are also other possibilities for analysis on the graph itself such as generating hierarchical representations of the data through reachabilityplot dendrograms and the hierarchy of clusters detected by the algorithm can be accessed through the clusterhierarchy parameter The plot above has been colorcoded so that cluster colors in planar space match the linear segment clusters of the reachability plot Note that the blue and red clusters are adjacent in the reachability plot and can be hierarchically represented as children of a larger parent cluster

Examples

- Demo of OPTICS clustering algorithm

Comparison with DBSCAN

The results from OPTICS clusteropticsdbscan method and DBSCAN are very similar but not always identical specifically labeling of periphery and noise points This is in part because the first samples of each dense area processed by OPTICS have a large reachability value while being close to other points in their area and will thus sometimes be marked as noise rather than periphery This affects adjacent points when they are considered as candidates for being marked as either periphery or noise

Note that for any single value of ϵ DBSCAN will tend to have a shorter run time than OPTICS however for repeated runs at varying ϵ values a single run of OPTICS may require less cumulative runtime than DBSCAN It is also important to note that OPTICS' output is close to DBSCAN's only if ϵ and $\max\epsilon$ are close

scikitlearn user guide Release 0213

Computational Complexity

Spatial indexing trees are used to avoid calculating the full distance matrix and allow for efficient memory usage on large sets of samples Different distance metrics can be supplied via the metric keyword

For large datasets similar but not identical results can be obtained via HDBSCAN The HDBSCAN implementation is multithreaded and has better algorithmic runtime complexity than OPTICS at the cost of worse memory scaling For extremely large datasets that exhaust system memory using HDBSCAN OPTICS will maintain a solution in a reasonable amount of wall time

References

- “OPTICS ordering points to identify the clustering structure” Ankerst Mihael Markus M Breunig Hans Peter Kriegel and Jörg Sander In ACM Sigmod Record vol 28 no 2 pp 4960 ACM 1999

Birch

The Birch builds a tree called the Characteristic Feature Tree CFT for the given data The data is essentially lossy compressed to a set of Characteristic Feature nodes CF Nodes The CF Nodes have a number of subclusters called Characteristic Feature subclusters CF Subclusters and these CF Subclusters located in the nonterminal CF Nodes can have CF Nodes as children

The CF Subclusters hold the necessary information for clustering which prevents the need to hold the entire input data in memory This information includes

- Number of samples in a subcluster
- Linear Sum A ndimensional vector holding the sum of all samples
- Squared Sum Sum of the squared L2 norm of all samples
- Centroids To avoid recalculation linear sum / nsamples
- Squared norm of the centroids

The Birch algorithm has two parameters the threshold and the branching factor The branching factor limits the number of subclusters in a node and the threshold limits the distance between the entering sample and the existing subclusters

This algorithm can be viewed as an instance or data reduction method since it reduces the input data to a set of subclusters which are obtained directly from the leaves of the CFT This reduced data can be further processed by feeding it into a global clusterer This global clusterer can be set by nclusters If nclusters is set to None the subclusters from the leaves are directly read off otherwise a global clustering step labels these subclusters into global clusters labels and the samples are mapped to the global label of the nearest subcluster

Algorithm description

- A new sample is inserted into the root of the CF Tree which is a CF Node It is then merged with the subcluster of the root that has the smallest radius after merging constrained by the threshold and branching factor conditions If the subcluster has any child node then this is done repeatedly till it reaches a leaf After finding the nearest subcluster in the leaf the properties of this subcluster and the parent subclusters are recursively updated
- If the radius of the subcluster obtained by merging the new sample and the nearest subcluster is greater than the square of the threshold and if the number of subclusters is greater than the branching factor then a space is temporarily allocated to this new sample The two farthest subclusters are taken and the subclusters are divided into two groups on the basis of the distance between these subclusters

scikitlearn user guide Release 0213

- If this split node has a parent subcluster and there is room for a new subcluster then the parent is split into two
- If there is no room then this node is again split into two and the process is continued recursively till it reaches the root

Birch or MiniBatchKMeans

- Birch does not scale very well to high dimensional data As a rule of thumb if nfeatures is greater than twenty it is generally better to use MiniBatchKMeans
- If the number of instances of data needs to be reduced or if one wants a large number of subclusters either as a preprocessing step or otherwise Birch is more useful than MiniBatchKMeans

How to use partialfit

To avoid the computation of global clustering for every call of partialfit the user is advised

- 1 To setnclustersNone initially
- 2 Train all data by multiple calls to partialfit
- 3 Setnclusters to a required value using brcsetparamsnclustersnclusters
- 4 Callpartialfit finally with no arguments ie brcpartialfit which performs the global clustering

References

- Tian Zhang Raghu Ramakrishnan Maron Livny BIRCH An efficient data clustering method for large databases <https://www.cs.fu.ca/Course/Central459/hanpapers/zhang96.pdf>
- Roberto Perdisci JBirch Java implementation of BIRCH clustering algorithm <https://code.google.com/archive/p/jbirch/>

Clustering performance evaluation

Evaluating the performance of a clustering algorithm is not as trivial as counting the number of errors or the precision and recall of a supervised classification algorithm In particular any evaluation metric should not take the absolute values of the cluster labels into account but rather if this clustering define separations of the data similar to some ground truth set of classes or satisfying some assumption such that members belong to the same class are more similar that members of different classes according to some similarity metric

scikitlearn user guide Release 0213

Adjusted Rand index

Given the knowledge of the ground truth class assignments `labelstrue` and our clustering algorithm assignments of the same samples `labelspred` the adjusted Rand index is a function that measures the similarity of the two assignments ignoring permutations and with chance normalization

`from sklearn import metrics`

`labelstrue 0 0 0 1 1 1`

`labelspred 0 0 1 1 2 2`

`metricsadjustedrandscorelabelstrue labelspred`
024

One can permute 0 and 1 in the predicted labels rename 2 to 3 and get the same score

`labelspred 1 1 0 0 3 3`

`metricsadjustedrandscorelabelstrue labelspred`
024

Furthermore `adjustedrandscore` issymmetric swapping the argument does not change the score It can thus be used as a consensus measure

`metricsadjustedrandscorelabelspred labelstrue`
024

Perfect labeling is scored 10

`labelspred labelstrue`

`metricsadjustedrandscorelabelstrue labelspred`
10

Bad eg independent labelings have negative or close to 00 scores

`labelstrue 0 1 2 0 3 4 5 1`

`labelspred 1 1 0 0 2 2 2 2`

`metricsadjustedrandscorelabelstrue labelspred`
012

Advantages

- Random uniform label assignments have a ARI score close to 00 for any value of `nclusters` and `nsamples` which is not the case for raw Rand index or the Vmeasure for instance
- Bounded range 1 1 negative values are bad independent labelings similar clusterings have a positive ARI 10 is the perfect match score
- No assumption is made on the cluster structure can be used to compare clustering algorithms such as k means which assumes isotropic blob shapes with results of spectral clustering algorithms which can find cluster with “folded” shapes

Drawbacks

- Contrary to inertia ARI requires knowledge of the ground truth classes while is almost never available in practice or requires manual assignment by human annotators as in the supervised learning setting
- 32 Unsupervised learning 373

scikitlearn user guide Release 0213

However ARI can also be useful in a purely unsupervised setting as a building block for a Consensus Index that can be used for clustering model selection TODO

Examples

- Adjustment for chance in clustering performance evaluation Analysis of the impact of the dataset size on the value of clustering measures for random assignments

Mathematical formulation

If C is a ground truth class assignment and K the clustering let us define n_{ii} and n_{ij} as

- n_{ii} the number of pairs of elements that are in the same set in C and in the same set in K
- n_{ij} the number of pairs of elements that are in different sets in C and in different sets in K

The raw unadjusted Rand index is then given by

$$RI = \frac{n_{ii} + \frac{1}{2} \sum_{i \neq j} n_{ij}}{\frac{N(N-1)}{2}}$$

Where $\frac{N(N-1)}{2}$

2 is the total number of possible pairs in the dataset without ordering

However the RI score does not guarantee that random label assignments will get a value close to zero esp if the number of clusters is in the same order of magnitude as the number of samples

To counter this effect we can discount the expected RI of random labelings by defining the adjusted Rand index as follows

$$ARI = \frac{RI - E[RI]}{\max RI - E[RI]}$$

References

- Comparing Partitions L Hubert and P Arabie Journal of Classification 1985
- Wikipedia entry for the adjusted Rand index

Mutual Information based scores

Given the knowledge of the ground truth class assignments labelstrue and our clustering algorithm assignments of the same samples labelspred the Mutual Information is a function that measures the agreement of the two assignments ignoring permutations Two different normalized versions of this measure are available Normalized Mutual Information NMI and Adjusted Mutual Information AMI NMI is often used in the literature while AMI was proposed more recently and is normalized against chance

from sklearn import metrics

labelstrue = [0, 0, 0, 1, 1, 1]

labelspred = [0, 0, 1, 1, 2, 2]

metrics.adjusted_mutual_info_score(labelstrue, labelspred)

0.22504

One can permute 0 and 1 in the predicted labels rename 2 to 3 and get the same score

374 Chapter 3 User Guide

scikitlearn user guide Release 0213

labelspred 1 1 0 0 3 3

metricsadjustedmutualinfoscorelabelstrue labelspred
022504

Allmutualinfoscore adjustedmutualinfoscore andnormalizedmutualinfoscore

are symmetric swapping the argument does not change the score Thus they can be used as a consensus measure

metricsadjustedmutualinfoscorelabelspred labelstrue
022504

Perfect labeling is scored 10

labelspred labelstrue

metricsadjustedmutualinfoscorelabelstrue labelspred
10

metricsnormalizedmutualinfoscorelabelstrue labelspred
10

This is not true for mutualinfoscore which is therefore harder to judge

metricsmutualinfoscorelabelstrue labelspred
069

Bad eg independent labelings have nonpositive scores

labelstrue 0 1 2 0 3 4 5 1

labelspred 1 1 0 0 2 2 2 2

metricsadjustedmutualinfoscorelabelstrue labelspred
010526

Advantages

- Random uniform label assignments have a AMI score close to 00 for any value of nclusters and nsamples which is not the case for raw Mutual Information or the Vmeasure for instance
- Upper bound of 1 Values close to zero indicate two label assignments that are largely independent while values close to one indicate significant agreement Further an AMI of exactly 1 indicates that the two label assignments are equal with or without permutation

Drawbacks

- Contrary to inertia MIbased measures require the knowledge of the ground truth classes while almost never available in practice or requires manual assignment by human annotators as in the supervised learning setting

However MIbased measures can also be useful in purely unsupervised setting as a building block for a Consensus Index that can be used for clustering model selection

- NMI and MI are not adjusted against chance

Examples

32 Unsupervised learning 375

•Adjustment for chance in clustering performance evaluation Analysis of the impact of the dataset size on the value of clustering measures for random assignments This example also includes the Adjusted Rand Index

Mathematical formulation

Assume two label assignments of the same N objects \mathcal{X} and \mathcal{Y} Their entropy is the amount of uncertainty for a partition set defined by

$$H(\mathcal{X}) = -\sum_{i=1}^K p_i \log p_i$$

where p_i is the probability that an object picked at random from \mathcal{X} falls into class i Likewise for \mathcal{Y}

With p_{ij} The mutual information MI between \mathcal{X} and \mathcal{Y} is calculated by

$$MI(\mathcal{X}, \mathcal{Y}) = \sum_{i=1}^K \sum_{j=1}^L p_{ij} \log \frac{p_{ij}}{p_i p_j}$$

where p_{ij} is the probability that an object picked at random falls into both classes i and j It also can be expressed in set cardinality formulation

$$MI(\mathcal{X}, \mathcal{Y}) = \sum_{i=1}^K \sum_{j=1}^L \frac{|S_{ij}|}{N} \log \frac{|S_{ij}|}{|S_i| |S_j|}$$

The normalized mutual information is defined as

$$NMI(\mathcal{X}, \mathcal{Y}) = \frac{MI(\mathcal{X}, \mathcal{Y})}{\frac{H(\mathcal{X}) + H(\mathcal{Y})}{2}}$$

mean \mathcal{X}, \mathcal{Y}

This value of the mutual information and also the normalized variant is not adjusted for chance and will tend to increase as the number of different labels clusters increases regardless of the actual amount of “mutual information” between the label assignments

The expected value for the mutual information can be calculated using the following equation VEB2009 In this equation n_X the number of elements in \mathcal{X} and n_Y the number of elements in \mathcal{Y}

$$E[MI(\mathcal{X}, \mathcal{Y})] = \sum_{i=1}^K \sum_{j=1}^L \frac{1}{n_X n_Y} \log \frac{n_X n_Y}{n_X n_Y} = 0$$
$$E[NMI(\mathcal{X}, \mathcal{Y})] = \frac{E[MI(\mathcal{X}, \mathcal{Y})]}{\frac{H(\mathcal{X}) + H(\mathcal{Y})}{2}} = 0$$

Using the expected value the adjusted mutual information can then be calculated using a similar form to that of the adjusted Rand index

$$AMI = \frac{MI - E[MI]}{\max(MI, 1 - E[MI])}$$

For normalized mutual information and adjusted mutual information the normalizing value is typically some generalized mean of the entropies of each clustering Various generalized means exist and no firm rules exist for preferring one over the others The decision is largely a fieldbyfield basis for instance in community detection the arithmetic mean is most common Each normalizing method provides “qualitatively similar behaviours” YAT2016 In our implementation this is controlled by the averagemethod parameter

Vinh et al 2010 named variants of NMI and AMI by their averaging method VEB2010 Their ‘sqrt’ and ‘sum’ averages are the geometric and arithmetic means we use these more broadly common names

scikitlearn user guide Release 0213

References

- Strehl Alexander and Joydeep Ghosh 2002 “Cluster ensembles – a knowledge reuse frame work for combining multiple partitions” Journal of Machine Learning Research 3 583–617 doi101162153244303321897735

- Wikipedia entry for the normalized Mutual Information
- Wikipedia entry for the Adjusted Mutual Information

Homogeneity completeness and Vmeasure

Given the knowledge of the ground truth class assignments of the samples it is possible to define some intuitive metric using conditional entropy analysis

In particular Rosenberg and Hirschberg 2007 define the following two desirable objectives for any cluster assignment

- homogeneity each cluster contains only members of a single class
- completeness all members of a given class are assigned to the same cluster

We can turn those concept as scores homogeneityscore andcompletenessscore Both are bounded below by 00 and above by 10 higher is better

from sklearn import metrics

labelstrue 0 0 0 1 1 1

labelspred 0 0 1 1 2 2

metricshomogeneityscorelabelstrue labelspred

066

metricscompletenessscorelabelstrue labelspred

042

Their harmonic mean called Vmeasure is computed by vmeasurescore

metricsvmeasurescorelabelstrue labelspred

051

This function’s formula is as follows

$$V = \frac{1}{\beta + 1} (\beta \times \text{homogeneity} + \text{completeness})$$

↪
$$\beta \times \text{homogeneity} + \text{completeness}$$

beta defaults to a value of 10 but for using a value less than 1 for beta

metricsvmeasurescorelabelstrue labelspred beta06

054

more weight will be attributed to homogeneity and using a value greater than 1

metricsvmeasurescorelabelstrue labelspred beta18

048

more weight will be attributed to completeness

32 Unsupervised learning 377

scikitlearn user guide Release 0213

The Vmeasure is actually equivalent to the mutual information NMI discussed above with the aggregation function being the arithmetic mean B2011

Homogeneity completeness and Vmeasure can be computed at once using

homogeneitycompletenessvmeasure as follows

metricshomogeneitycompletenessvmeasurelabelstrue labelspred

066 042 051

The following clustering assignment is slightly better since it is homogeneous but not complete

labelspred 0 0 0 1 2 2

metricshomogeneitycompletenessvmeasurelabelstrue labelspred

10 068 081

Notevmeasurescore issymmetric it can be used to evaluate the agreement of two independent assignments on the same dataset

This is not the case for completenessscore andhomogeneityscore both are bound by the relationship

homogeneityscorea b completenessscoreb a

Advantages

- Bounded scores 00 is as bad as it can be 10 is a perfect score
- Intuitive interpretation clustering with bad Vmeasure can be qualitatively analyzed in terms of homogeneity and completeness to better feel what ‘kind’ of mistakes is done by the assignment
- No assumption is made on the cluster structure can be used to compare clustering algorithms such as k means which assumes isotropic blob shapes with results of spectral clustering algorithms which can find cluster with “folded” shapes

Drawbacks

- The previously introduced metrics are not normalized with regards to random labeling this means that depending on the number of samples clusters and ground truth classes a completely random labeling will not always yield the same values for homogeneity completeness and hence vmeasure In particular random labeling won’t yield zero scores especially when the number of clusters is large

This problem can safely be ignored when the number of samples is more than a thousand and the number of clusters is less than 10 For smaller sample sizes or larger number of clusters it is safer to use an adjusted index such as the Adjusted Rand Index ARI

- These metrics require the knowledge of the ground truth classes while almost never available in practice or requires manual assignment by human annotators as in the supervised learning setting

Examples

- Adjustment for chance in clustering performance evaluation Analysis of the impact of the dataset size on the value of clustering measures for random assignments

378 Chapter 3 User Guide

Mathematical formulation

Homogeneity and completeness scores are formally given by

$$h = 1 - \frac{H(C|K)}{H(K)}$$

$$c = 1 - \frac{H(K|C)}{H(K)}$$

$$V = \frac{2}{h + c}$$

where $H(K|C)$ is the conditional entropy of the classes given the cluster assignments and is given by

$$H(K|C) = -\sum_{k=1}^K \sum_{c=1}^C \frac{n_{kc}}{n} \log \frac{n_{kc}}{n}$$

$$H(K) = -\sum_{k=1}^K \frac{n_k}{n} \log \frac{n_k}{n}$$

$$H(C) = -\sum_{c=1}^C \frac{n_c}{n} \log \frac{n_c}{n}$$

$$H(K, C) = -\sum_{k=1}^K \sum_{c=1}^C \frac{n_{kc}}{n} \log \frac{n_{kc}}{n}$$

$$H(K, C) = -\sum_{k=1}^K \sum_{c=1}^C \frac{n_{kc}}{n} \log \frac{n_{kc}}{n}$$

and $H(K)$ is the entropy of the classes and is given by

$$H(K) = -\sum_{k=1}^K \frac{n_k}{n} \log \frac{n_k}{n}$$

$$H(C) = -\sum_{c=1}^C \frac{n_c}{n} \log \frac{n_c}{n}$$

$$H(K, C) = -\sum_{k=1}^K \sum_{c=1}^C \frac{n_{kc}}{n} \log \frac{n_{kc}}{n}$$

$$H(K, C) = -\sum_{k=1}^K \sum_{c=1}^C \frac{n_{kc}}{n} \log \frac{n_{kc}}{n}$$

with n the total number of samples n_k and n_c the number of samples respectively belonging to class k and cluster c

and finally n_{kc} the number of samples from class k assigned to cluster c

The conditional entropy of clusters given class k and the entropy of clusters C are defined in a symmetric manner

Rosenberg and Hirschberg further define Vmeasure as the harmonic mean of homogeneity and completeness

$$V = \frac{2}{h + c}$$

$$h = \frac{H(K|C)}{H(K)}$$

References

- VMeasure A conditional entropybased external cluster evaluation measure Andrew Rosenberg and Julia

Hirschberg 2007

FowlkesMallows scores

The FowlkesMallows index `sklearn.metrics.fowlkesmallowsscore` can be used when the ground

truth class assignments of the samples is known The FowlkesMallows score FMI is defined as the geometric mean

of the pairwise precision and recall

FMI = $\sqrt{\frac{TP}{(TP + FP) * (TP + FN)}}$

TPFPFPFN

Where TP is the number of True Positive ie the number of pair of points that belong to the same clusters in both the

true labels and the predicted labels FP is the number of False Positive ie the number of pair of points that belong

to the same clusters in the true labels and not in the predicted labels and FN is the number of False Negative ie the

number of pair of points that belongs in the same clusters in the predicted labels and not in the true labels

The score ranges from 0 to 1 A high value indicates a good similarity between two clusters

from sklearn import metrics

labelstrue 0 0 0 1 1 1

labelspred 0 0 1 1 2 2

scikitlearn user guide Release 0213

metricsfowlkesmallowsscorelabelstrue labelspred
047140

One can permute 0 and 1 in the predicted labels rename 2 to 3 and get the same score

labelspred 1 1 0 0 3 3

metricsfowlkesmallowsscorelabelstrue labelspred
047140

Perfect labeling is scored 10

labelspred labelstrue

metricsfowlkesmallowsscorelabelstrue labelspred
10

Bad eg independent labelings have zero scores

labelstrue 0 1 2 0 3 4 5 1

labelspred 1 1 0 0 2 2 2 2

metricsfowlkesmallowsscorelabelstrue labelspred
00

Advantages

- Random uniform label assignments have a FMI score close to 00 for any value of nclusters and nsamples which is not the case for raw Mutual Information or the Vmeasure for instance
- Upperbounded at 1 Values close to zero indicate two label assignments that are largely independent while values close to one indicate significant agreement Further values of exactly 0 indicate purely independent label assignments and a FMI of exactly 1 indicates that the two label assignments are equal with or without permutation
- No assumption is made on the cluster structure can be used to compare clustering algorithms such as k means which assumes isotropic blob shapes with results of spectral clustering algorithms which can find cluster with “folded” shapes

Drawbacks

- Contrary to inertia FMIbased measures require the knowledge of the ground truth classes while almost never available in practice or requires manual assignment by human annotators as in the supervised learning setting

References

- E B Fowlkes and C L Mallows 1983 “A method for comparing two hierarchical clusterings” Journal of the American Statistical Association <http://wildfire.statucla.edu/pdf/library/fowlkes.pdf>
- Wikipedia entry for the FowlkesMallows Index

Silhouette Coefficient

If the ground truth labels are not known evaluation must be performed using the model itself The Silhouette Coefficient `sklearn.metrics.silhouette_score` is an example of such an evaluation where a higher Silhouette Coefficient score relates to a model with better defined clusters The Silhouette Coefficient is defined for each sample and is composed of two scores

- a The mean distance between a sample and all other points in the same class
- b The mean distance between a sample and all other points in the next nearest cluster

The Silhouette Coefficient `s` for a single sample is then given as

$$s = \frac{b - a}{\max(b, a)}$$

The Silhouette Coefficient for a set of samples is given as the mean of the Silhouette Coefficient for each sample

```
from sklearn import metrics
from sklearn.metrics import pairwise_distances
from sklearn import datasets
dataset = datasets.load_iris()
X = dataset.data
y = dataset.target
```

In normal usage the Silhouette Coefficient is applied to the results of a cluster analysis

```
import numpy as np
from sklearn.cluster import KMeans
kmeans_model = KMeans(n_clusters=3, random_state=1)
fit_X = kmeans_model.fit(X)
labels = kmeans_model.labels_
metricssilhouette_score(X, labels, metric=euclidean)
```

References

- Peter J Rousseeuw 1987 “Silhouettes a Graphical Aid to the Interpretation and Validation of Cluster Analysis” Computational and Applied Mathematics 20 53-65 doi10.1016/0377042787901257

Advantages

- The score is bounded between -1 for incorrect clustering and 1 for highly dense clustering Scores around zero indicate overlapping clusters
- The score is higher when clusters are dense and well separated which relates to a standard concept of a cluster

Drawbacks

- The Silhouette Coefficient is generally higher for convex clusters than other concepts of clusters such as density based clusters like those obtained through DBSCAN

Examples

•Selecting the number of clusters with silhouette analysis on KMeans clustering In this example the silhouette analysis is used to choose an optimal value for nclusters

CalinskiHarabasz Index

If the ground truth labels are not known the CalinskiHarabasz index sklearnmetrics.calinskiharabaszscore also known as the Variance Ratio Criterion can be used to evaluate the model where a higher CalinskiHarabasz score relates to a model with better defined clusters

For clusters the CalinskiHarabasz score is given as the ratio of the betweenclusters dispersion mean and the withincluster dispersion

$$\frac{\text{Tr}(B)}{\text{Tr}(W)} \times \frac{1}{k-1}$$
where B is the between group dispersion matrix and W is the withincluster dispersion matrix defined by

$$B = \frac{1}{k} \sum_{k=1}^k (\bar{x}_k - \bar{x})(\bar{x}_k - \bar{x})^T$$
$$W = \sum_{k=1}^k \sum_{i \in C_k} (x_i - \bar{x}_k)(x_i - \bar{x}_k)^T$$
with n be the number of points in our data C be the set of points in cluster C be the center of cluster C be the center of C n be the number of points in cluster C

```
from sklearn import metrics
from sklearnmetrics import pairwisedistances
from sklearn import datasets
dataset = datasets.loadiris
X = dataset.data
y = dataset.target
In normal usage the CalinskiHarabasz index is applied to the results of a cluster analysis
import numpy as np
from sklearncluster import KMeans
kmeansmodel = KMeans(nclusters=3, randomstate=1).fit(X)
labels = kmeansmodel.labels
metrics.calinskiharabaszscore(X, labels)
56162
```

Advantages

- The score is higher when clusters are dense and well separated which relates to a standard concept of a cluster
- The score is fast to compute

Drawbacks

- The CalinskiHarabasz index is generally higher for convex clusters than other concepts of clusters such as density based clusters like those obtained through DBSCAN

References

- Cali ´nski T Harabasz J 1974 “A dendrite method for cluster analysis” Communications in Statistics theory and Methods 3 127 doi101080036109262011560741

DaviesBouldin Index

If the ground truth labels are not known the DaviesBouldin index sklearnmetrics

daviesbouldinscore can be used to evaluate the model where a lower DaviesBouldin index relates to a model with better separation between the clusters

The index is defined as the average similarity between each cluster μ_i and its most similar one μ_j In the context of this index similarity is defined as a measure $d(\mu_i, \mu_j)$ that trades off

- $d(\mu_i, \mu_j)$ the average distance between each point of cluster μ_i and the centroid of that cluster – also know as cluster diameter
- $d(\mu_i, \mu_j)$ the distance between cluster centroids μ_i and μ_j

A simple choice to construct $d(\mu_i, \mu_j)$ so that it is nonnegative and symmetric is

$$d(\mu_i, \mu_j) = \frac{1}{n_i} \sum_{x \in \mu_i} \|x - \mu_i\| + \frac{1}{n_j} \sum_{x \in \mu_j} \|x - \mu_j\|$$

Then the DaviesBouldin index is defined as

$$DB = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \frac{d(\mu_i, \mu_j)}{\min(d(\mu_i, \mu_j), d(\mu_j, \mu_i))}$$

Zero is the lowest possible score Values closer to zero indicate a better partition

In normal usage the DaviesBouldin index is applied to the results of a cluster analysis as follows

```
from sklearn import datasets
iris = datasets.loadiris()
X = iris.data
from sklearncluster import KMeans
from sklearnmetrics import daviesbouldinscore
kmeans = KMeans(nclusters=3, randomstate=1)
fitX = kmeans.fit(X)
labels = kmeans.labels_
daviesbouldinscore(X, labels)
```

Advantages

- The computation of DaviesBouldin is simpler than that of Silhouette scores
- The index is computed only quantities and features inherent to the dataset

Drawbacks

- The DaviesBoulding index is generally higher for convex clusters than other concepts of clusters such as density based clusters like those obtained from DBSCAN
- The usage of centroid distance limits the distance metric to Euclidean space
- A good value reported by this method does not imply the best information retrieval

References

- Davies David L Bouldin Donald W 1979 “A Cluster Separation Measure” IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI1 2 224227 doi101109TPAMI19794766909
- Halkidi Maria Batistakis Yannis Vazirgiannis Michalis 2001 “On Clustering Validation Techniques” Journal of Intelligent Information Systems 1723 107145 doi101023A1012801612483
- Wikipedia entry for DaviesBouldin index

Contingency Matrix

Contingency matrix sklearnmetricsclustercontingencymatrix reports the intersection cardinality for every truepredicted cluster pair The contingency matrix provides sufficient statistics for all clustering metrics where the samples are independent and identically distributed and one doesn’t need to account for some instances not being clustered

Here is an example

```
from sklearnmetricscluster import contingencymatrix
x = ['a', 'a', 'b', 'b', 'b']
y = [0, 0, 1, 1, 2]
contingencymatrix(x, y)
array([[2, 1, 0],
       [0, 1, 2]])
```

The first row of output array indicates that there are three samples whose true cluster is “a” Of them two are in predicted cluster 0 one is in 1 and none is in 2 And the second row indicates that there are three samples whose true cluster is “b” Of them none is in predicted cluster 0 one is in 1 and two are in 2

A confusion matrix for classification is a square contingency matrix where the order of rows and columns correspond to a list of classes

Advantages

- Allows to examine the spread of each true cluster across predicted clusters and vice versa
- The contingency table calculated is typically utilized in the calculation of a similarity statistic like the others listed in this document between the two clusterings

Drawbacks

- Contingency matrix is easy to interpret for a small number of clusters but becomes very hard to interpret for a large number of clusters

scikitlearn user guide Release 0213

- It doesn't give a single metric to use as an objective for clustering optimisation

References

- Wikipedia entry for contingency matrix

324 Biclustering

Biclustering can be performed with the module `sklearn.cluster.bicluster`. Biclustering algorithms simultaneously cluster rows and columns of a data matrix. These clusters of rows and columns are known as biclusters. Each determines a submatrix of the original data matrix with some desired properties.

For instance, given a matrix of shape 10 x 10, one possible bicluster with three rows and two columns induces a submatrix of shape 3 x 2:

```
import numpy as np
data = np.arange(100).reshape(10, 10)
rows = np.array([0, 2, 3])
columns = np.array([1, 2])
data[rows, columns]
array([1, 2, 3, 12, 13, 14, 22, 23, 24, 32, 33, 34])
```

For visualization purposes, given a bicluster, the rows and columns of the data matrix may be rearranged to make the bicluster contiguous.

Algorithms differ in how they define biclusters. Some of the common types include:

- constant values (constant rows or constant columns)
- unusually high or low values
- submatrices with low variance
- correlated rows or columns

Algorithms also differ in how rows and columns may be assigned to biclusters, which leads to different bicluster structures. Block diagonal or checkerboard structures occur when rows and columns are divided into partitions. If each row and each column belongs to exactly one bicluster, then rearranging the rows and columns of the data matrix reveals the biclusters on the diagonal. Here is an example of this structure where biclusters have higher average values than the other rows and columns:

In the checkerboard case, each row belongs to all column clusters and each column belongs to all row clusters. Here is an example of this structure where the variance of the values within each bicluster is small:

After fitting a model, row and column cluster membership can be found in the `rows` and `columns` attributes:

`rows[i]` is a binary vector with nonzero entries corresponding to rows that belong to bicluster `i`. Similarly, `columns[i]` indicates which columns belong to bicluster `i`.

Some models also have `row_labels` and `column_labels` attributes. These models partition the rows and columns such as in the block diagonal and checkerboard bicluster structures.

Note: Biclustering has many other names in different fields, including `coclustering`, `two-mode clustering`, `two-way clustering`, `block clustering`, `coupled two-way clustering`, etc. The names of some algorithms, such as the Spectral CoClustering algorithm, reflect these alternate names.

scikitlearn user guide Release 0213

Fig 35 An example of biclusters formed by partitioning rows and columns

Fig 36 An example of checkerboard biclusters

32 Unsupervised learning 387

Spectral CoClustering

TheSpectralCoclustering algorithm finds biclusters with values higher than those in the corresponding other rows and columns Each row and each column belongs to exactly one bicluster so rearranging the rows and columns to make partitions contiguous reveals these high values along the diagonal

Note The algorithm treats the input data matrix as a bipartite graph the rows and columns of the matrix correspond to the two sets of vertices and each entry corresponds to an edge between a row and a column The algorithm approximates the normalized cut of this graph to find heavy subgraphs

Mathematical formulation

An approximate solution to the optimal normalized cut may be found via the generalized eigenvalue decomposition of the Laplacian of the graph Usually this would mean working directly with the Laplacian matrix If the original data matrix X has shape $n \times m$ the Laplacian matrix for the corresponding bipartite graph has shape $(n+m) \times (n+m)$ However in this case it is possible to work directly with X which is smaller and more efficient

The input matrix X is preprocessed as follows

$$X \leftarrow \frac{X - 12X}{\sqrt{12}}$$

Where I is the diagonal matrix with entry 1 equal to Σ

$X \leftarrow X - \frac{XX^T X}{\text{trace}(XX^T)}$ and I is the diagonal matrix with entry 1 equal to Σ

$X \leftarrow X - \frac{XX^T X}{\text{trace}(XX^T)}$

The singular value decomposition $U \Sigma V^T$ provides the partitions of the rows and columns of X A subset of the left singular vectors gives the row partitions and a subset of the right singular vectors gives the column partitions The $\lfloor \log_2 n \rfloor$ singular vectors starting from the second provide the desired partitioning information They are used to form the matrix U

$$U = \begin{bmatrix} I & 0 \\ 0 & -12I \\ 0 & -12I \end{bmatrix}$$

where the columns of U are $n/2$ $n/2$ $n/2$ and similarly for V

Then the rows of U are clustered using kmeans The first $n/2$ rows labels provide the row partitioning and the remaining $n/2$ columns labels provide the column partitioning

Examples

- A demo of the Spectral CoClustering algorithm A simple example showing how to generate a data matrix with biclusters and apply this method to it
- Biclustering documents with the Spectral Coclustering algorithm An example of finding biclusters in the twenty newsgroup dataset

References

- Dhillon Inderjit S 2001 Coclustering documents and words using bipartite spectral graph partitioning

Spectral Biclustering

TheSpectralBiclustering algorithm assumes that the input data matrix has a hidden checkerboard structure The rows and columns of a matrix with this structure may be partitioned so that the entries of any bicluster in the Cartesian product of row clusters and column clusters are approximately constant For instance if there are two row partitions and three column partitions each row will belong to three biclusters and each column will belong to two biclusters

The algorithm partitions the rows and columns of a matrix so that a corresponding blockwiseconstant checkerboard matrix provides a good approximation to the original matrix

Mathematical formulation

The input matrix X is first normalized to make the checkerboard pattern more obvious There are three possible methods

1Independent row and column normalization as in Spectral CoClustering This method makes the rows sum to a constant and the columns sum to a different constant

2Bistochastization repeated row and column normalization until convergence This method makes both rows and columns sum to the same constant

3Log normalization the log of the data matrix is computed $\log X$ Then the column mean \bar{c} row mean \bar{r} and overall mean \bar{m} of X are computed The final matrix is computed according to the formula

$$\frac{X - \bar{c} - \bar{r} + \bar{m}}{\bar{c} \bar{r}}$$

After normalizing the first few singular vectors are computed just as in the Spectral CoClustering algorithm If log normalization was used all the singular vectors are meaningful However if independent normalization or bistochastization were used the first singular vectors v_1 and w_1 are discarded From now on the “first” singular vectors refers to v_2 w_1 and v_2 w_1 except in the case of log normalization

Given these singular vectors they are ranked according to which can be best approximated by a piecewiseconstant vector The approximations for each vector are found using onedimensional kmeans and scored using the Euclidean distance Some subset of the best left and right singular vector are selected Next the data is projected to this best subset of singular vectors and clustered

For instance if k singular vectors were calculated the k best are found as described where $k = 2$ Let X be the matrix with columns the k best left singular vectors and similarly Y for the right To partition the rows the rows of X are projected to a k dimensional space XY Treating the k rows of this $k \times k$ matrix as samples and clustering using kmeans yields the row labels Similarly projecting the columns to $Y^T X$ and clustering this $k \times k$ matrix yields the column labels

Examples

•A demo of the Spectral Biclustering algorithm a simple example showing how to generate a checkerboard matrix and bicluster it

References

- Kluger Yuval et al 2003 Spectral biclustering of microarray data coclustering genes and conditions

Biclustering evaluation

There are two ways of evaluating a biclustering result internal and external Internal measures such as cluster stability rely only on the data and the result themselves Currently there are no internal bicluster measures in scikit learn External measures refer to an external source of information such as the true solution When working with real data the true solution is usually unknown but biclustering artificial data may be useful for evaluating algorithms precisely because the true solution is known

To compare a set of found biclusters to the set of true biclusters two similarity measures are needed a similarity measure for individual biclusters and a way to combine these individual similarities into an overall score

To compare individual biclusters several measures have been used For now only the Jaccard index is implemented

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

$$J(A, B) = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

where A and B are biclusters $|A \cap B|$ is the number of elements in their intersection The Jaccard index achieves its minimum of 0 when the biclusters do not overlap at all and its maximum of 1 when they are identical

Several methods have been developed to compare two sets of biclusters For now only consensus_score Hochreiter et al 2010 is available

1 Compute bicluster similarities for pairs of biclusters one in each set using the Jaccard index or a similar measure

2 Assign biclusters from one set to another in a one-to-one fashion to maximize the sum of their similarities This step is performed using the Hungarian algorithm

3 The final sum of similarities is divided by the size of the larger set

The minimum consensus score 0 occurs when all pairs of biclusters are totally dissimilar The maximum score 1 occurs when both sets are identical

References

- Hochreiter Bodenhofer et al 2010 FABIA factor analysis for bicluster acquisition

325 Decomposing signals in components matrix factorization problems

Principal component analysis PCA (Principal Component Analysis)

Exact PCA (Principal Component Analysis) and probabilistic interpretation

PCA (Principal Component Analysis) is used to decompose a multivariate dataset in a set of successive orthogonal components that explain a maximum amount of the variance In scikitlearn PCA (Principal Component Analysis) is implemented as a transformer object that learns k components in its fit method and can be used on new data to project it on these components

PCA (Principal Component Analysis) centers but does not scale the input data for each feature before applying the SVD The optional parameter `whiten=True` makes it possible to project the data onto the singular space while scaling each component to unit variance This is often useful if the models downstream make strong assumptions on the isotropy of the signal

this is for example the case for Support Vector Machines with the RBF (Radial Basis Function) kernel and the KMeans clustering algorithm Below is an example of the iris dataset which is comprised of 4 features projected on the 2 dimensions that explain most variance

The PCA object also provides a probabilistic interpretation of the PCA (Principal Component Analysis) that can give a likelihood of data based on the amount of variance it explains As such it implements a score method that can be used in crossvalidation

Examples

- Comparison of LDA and PCA (Principal Component Analysis) 2D projection of Iris dataset
- Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA

Incremental PCA (Principal Component Analysis)

ThePCA object is very useful but has certain limitations for large datasets The biggest limitation is that PCA (Principal Component Analysis) only supports batch processing which means all of the data to be processed must fit in main memory The IncrementalPCA object uses a different form of processing and allows for partial computations which almost exactly match the results ofPCA while processing the data in a minibatch fashion IncrementalPCA makes it possible to implement outof core Principal Component Analysis either by

- Using its partialfit method on chunks of data fetched sequentially from the local hard drive or a network database
- Calling its fit method on a memory mapped file using numpymemmap

IncrementalPCA only stores estimates of component and noise variances in order update explainedvarianceratio incrementally This is why memory usage depends on the number of samples per batch rather than the number of samples to be processed in the dataset

As inPCAIncrementalPCA centers but does not scale the input data for each feature before applying the SVD

Examples

- Incremental PCA (Principal Component Analysis)

PCA (Principal Component Analysis) using randomized SVD

It is often interesting to project data to a lowerdimensional space that preserves most of the variance by dropping the singular vector of components associated with lower singular values

For instance if we work with 64x64 pixel graylevel pictures for face recognition the dimensionality of the data is 4096 and it is slow to train an RBF (Radial Basis Function) support vector machine on such wide data Furthermore we know that the intrinsic dimensionality of the data is much lower than 4096 since all pictures of human faces look somewhat alike The samples lie on a manifold of much lower dimension say around 200 for instance The PCA (Principal Component Analysis) algorithm can be used to linearly transform the data while both reducing the dimensionality and preserve most of the explained variance at the same time

The classPCA used with the optional parameter svdsolverrandomized is very useful in that case since we are going to drop most of the singular vectors it is much more efficient to limit the computation to an approximated estimate of the singular vectors we will keep to actually perform the transform

For instance the following shows 16 sample portraits centered around 00 from the Olivetti dataset On the right hand side are the first 16 singular vectors reshaped as portraits Since we only require the top 16 singular vectors of a dataset with size 400 and 64x64 4096 the computation time is less than 1s

scikitlearn user guide Release 0213

If we note $\max \max \text{samples} \text{ features}$ and $\min \min \text{samples} \text{ features}$ the time complexity of the randomizedPCA is $O(n^2)$

$\max \cdot \text{components}$ instead of $O(n^2)$

$\max \cdot \min$ for the exact method implemented in PCA (Principal Component Analysis)

The memory footprint of randomized PCA (Principal Component Analysis) is also proportional to $2 \cdot \max \cdot \text{components}$ instead of $\max \cdot \min$ for the exact method

Note the implementation of `inverse_transform` in PCA with `svd_solver='randomized'` is not the exact

inverse transform of `transform` even when `whiten=False` default

Examples

- Faces recognition example using eigenfaces and SVMs
- Faces dataset decompositions

References

- “Finding structure with randomness Stochastic algorithms for constructing approximate matrix decompositions” Halko et al 2009

Kernel PCA (Principal Component Analysis)

KernelPCA is an extension of PCA (Principal Component Analysis) which achieves nonlinear dimensionality reduction through the use of kernels see Pairwise metrics Affinities and Kernels It has many applications including denoising compression and structured prediction kernel dependency estimation KernelPCA supports both transform and inversetransform

Examples

- Kernel PCA (Principal Component Analysis)

Sparse principal components analysis SparsePCA and MiniBatchSparsePCA

SparsePCA is a variant of PCA (Principal Component Analysis) with the goal of extracting the set of sparse components that best reconstruct the data

Minibatch sparse PCA (Principal Component Analysis) MiniBatchSparsePCA is a variant of SparsePCA that is faster but less accurate The increased speed is reached by iterating over small chunks of the set of features for a given number of iterations

Principal component analysis PCA (Principal Component Analysis) has the disadvantage that the components extracted by this method have exclusively dense expressions ie they have nonzero coefficients when expressed as linear combinations of the original variables This can make interpretation difficult In many cases the real underlying components can be more naturally imagined as sparse vectors for example in face recognition components might naturally map to parts of faces

Sparse principal components yields a more parsimonious interpretable representation clearly emphasizing which of the original features contribute to the differences between samples

The following example illustrates 16 components extracted using sparse PCA (Principal Component Analysis) from the Olivetti faces dataset It can be seen how the regularization term induces many zeros Furthermore the natural structure of the data causes the nonzero coefficients to be vertically adjacent The model does not enforce this mathematically each component is

scikitlearn user guide Release 0213

a vector $h \in \mathbb{R}^{4096}$ and there is no notion of vertical adjacency except during the humanfriendly visualization as 64x64 pixel images The fact that the components shown below appear local is the effect of the inherent structure of the data which makes such local patterns minimize reconstruction error There exist sparsityinducing norms that take into account adjacency and different kinds of structure see Jen09 for a review of such methods For more details on how to use Sparse PCA (Principal Component Analysis) see the Examples section below

Note that there are many different formulations for the Sparse PCA (Principal Component Analysis) problem The one implemented here is based on MrI09 The optimization problem solved is a PCA (Principal Component Analysis) problem dictionary learning with an ℓ_1 penalty on the components

$$\begin{aligned} & \arg \min_{W, A} \|W\|_1 \\ & \|W\|_1 - \|A\|_2^2 \\ & \|A\|_1 \end{aligned}$$

subject to $\|A\|_2 \leq 1$ for all $0 \leq i \leq n$

The sparsityinducing ℓ_1 norm also prevents learning components from noise when few training samples are available The degree of penalization and thus sparsity can be adjusted through the hyperparameter alpha Small values lead to a gently regularized factorization while larger values shrink many coefficients to zero

Note While in the spirit of an online algorithm the class MiniBatchSparsePCA does not implement partialfit because the algorithm is online along the features direction not the samples direction

Examples

- Faces dataset decompositions

References

Truncated singular value decomposition and latent semantic analysis

TruncatedSVD implements a variant of singular value decomposition SVD that only computes the σ largest singular values where σ is a userspecified parameter

When truncated SVD is applied to termdocument matrices as returned by CountVectorizer or

TfidfVectorizer this transformation is known as latent semantic analysis LSA because it transforms such

matrices to a “semantic” space of low dimensionality In particular LSA is known to combat the effects of synonymy

and polysemy both of which roughly mean there are multiple meanings per word which cause termdocument ma

trices to be overly sparse and exhibit poor similarity under measures such as cosine similarity

Note LSA is also known as latent semantic indexing LSI though strictly that refers to its use in persistent indexes

for information retrieval purposes

Mathematically truncated SVD applied to training samples X produces a lowrank approximation \hat{X}

$$\hat{X} \approx U \Sigma V^T$$

U

After this operation $U \Sigma V^T$

\hat{X} is the transformed training set with σ features called ncomponents in the API (Application Programming Interface)

To also transform a test set X_{test} we multiply it with U

$$\hat{X}_{test} = U X_{test}$$

Note Most treatments of LSA in the natural language processing NLP and information retrieval IR literature

swap the axes of the matrix X so that it has shape nfeatures x nsamples We present LSA in a different way

that matches the scikitlearn API (Application Programming Interface) better but the singular values found are the same

TruncatedSVD is very similar to PCA (Principal Component Analysis) but differs in that it works on sample matrices X directly instead of their

covariance matrices When the columnwise perfeature means of X are subtracted from the feature values truncated

SVD on the resulting matrix is equivalent to PCA (Principal Component Analysis) In practical terms this means that the TruncatedSVD transformer

acceptsscipysparse matrices without the need to densify them as densifying may fill up memory even for

mediumsized document collections

While theTruncatedSVD transformer works with any sparse feature matrix using it on tf-idf matrices is recom

mended over raw frequency counts in an LSAdocument processing setting In particular sublinear scaling and inverse

document frequency should be turned on sublineartf=True useidf=True to bring the feature values

closer to a Gaussian distribution compensating for LSA’s erroneous assumptions about textual data

Examples

- Clustering text documents using kmeans

References

- Christopher D Manning Prabhakar Raghavan and Hinrich Schütze 2008 Introduction to Information Retrieval Cambridge University Press chapter 18 Matrix decompositions latent semantic indexing

Dictionary Learning

Sparse coding with a precomputed dictionary

TheSparseCoder object is an estimator that can be used to transform signals into sparse linear combination of atoms from a fixed precomputed dictionary such as a discrete wavelet basis This object therefore does not implement afit method The transformation amounts to a sparse coding problem finding a representation of the data as a linear combination of as few dictionary atoms as possible All variations of dictionary learning implement the following transform methods controllable via the transformmethod initialization parameter

- Orthogonal matching pursuit Orthogonal Matching Pursuit OMP
- Leastangle regression Least Angle Regression
- Lasso computed by leastangle regression
- Lasso using coordinate descent Lasso
- Thresholding

Thresholding is very fast but it does not yield accurate reconstructions They have been shown useful in literature for classification tasks For image reconstruction tasks orthogonal matching pursuit yields the most accurate unbiased reconstruction

The dictionary learning objects offer via the splitcode parameter the possibility to separate the positive and negative values in the results of sparse coding This is useful when dictionary learning is used for extracting features that will be used for supervised learning because it allows the learning algorithm to assign different weights to negative loadings of a particular atom from to the corresponding positive loading

The split code for a single sample has length 2ncomponents and is constructed using the following rule First the regular code of length ncomponents is computed Then the first ncomponents entries of the splitcode are filled with the positive part of the regular code vector The second half of the split code is filled with the negative part of the code vector only with a positive sign Therefore the splitcode is nonnegative

Examples

- Sparse coding with a precomputed dictionary

Generic dictionary learning

Dictionary learning DictionaryLearning is a matrix factorization problem that amounts to finding a usually overcomplete dictionary that will perform well at sparsely encoding the fitted data

Representing data as sparse combinations of atoms from an overcomplete dictionary is suggested to be the way the mammalian primary visual cortex works Consequently dictionary learning applied on image patches has been shown to give good results in image processing tasks such as image completion inpainting and denoising as well as for supervised recognition tasks

Dictionary learning is an optimization problem solved by alternatively updating the sparse code as a solution to multiple Lasso problems considering the dictionary fixed and then updating the dictionary to best fit the sparse code

$$\begin{aligned} & \arg \min_{\mathbf{C}} \|\mathbf{C}\|_1 \\ & \frac{1}{2} \|\mathbf{X} - \mathbf{D}\mathbf{C}\|_2^2 \\ & \text{subject to } \|\mathbf{c}_i\|_2 \leq 1 \text{ for all } 0 \leq i < \text{atoms} \end{aligned}$$

After using such a procedure to fit the dictionary the transform is simply a sparse coding step that shares the same implementation with all dictionary learning objects see Sparse coding with a precomputed dictionary
It is also possible to constrain the dictionary and/or code to be positive to match constraints that may be present in the data Below are the faces with different positivity constraints applied Red indicates negative values blue indicates positive values and white represents zeros

scikitlearn user guide Release 0213

The following image shows how a dictionary learned from 4x4 pixel image patches extracted from part of the image of a raccoon face looks like

Examples

- Image denoising using dictionary learning

References

- “Online dictionary learning for sparse coding” J Mairal F Bach J Ponce G Sapiro 2009

Minibatch dictionary learning

MiniBatchDictionaryLearning implements a faster but less accurate version of the dictionary learning algorithm that is better suited for large datasets

By default MiniBatchDictionaryLearning divides the data into minibatches and optimizes in an online manner by cycling over the minibatches for the specified number of iterations However at the moment it does not implement a stopping condition

scikitlearn user guide Release 0213

The estimator also implements `partialfit` which updates the dictionary by iterating only once over a minibatch. This can be used for online learning when the data is not readily available from the start or for when the data does not fit into the memory.

Clustering for dictionary learning

Note that when using dictionary learning to extract a representation eg for sparse coding clustering can be a good proxy to learn the dictionary. For instance the `MiniBatchKMeans` estimator is computationally efficient and implements online learning with a `partialfit` method.

Example Online learning of a dictionary of parts of faces

Factor Analysis

In unsupervised learning we only have a dataset $X \in \mathbb{R}^{n \times 2}$. How can this dataset be described mathematically? A very simple continuous latent variable model for X is

$$X = h\epsilon$$

The vector h is called “latent” because it is unobserved. ϵ is considered a noise term distributed according to a Gaussian with mean 0 and covariance Σ . μ is some arbitrary offset vector. Such a model is called “generative” as it describes how X is generated from h . If we use all the h ’s as columns to form a matrix H and all the ϵ ’s as columns of a matrix E then we can write with suitably defined M and ϵ

$$X = H\epsilon$$

In other words we decomposed matrix X

If h is given the above equation automatically implies the following probabilistic interpretation

$$X = h\epsilon$$

For a complete probabilistic model we also need a prior distribution for the latent variable h . The most straightforward assumption based on the nice properties of the Gaussian distribution is $h \sim \mathcal{N}(0, I)$. This yields a Gaussian as the marginal distribution of X

$$X \sim \mathcal{N}(\mu, \Sigma)$$

Now without any further assumptions the idea of having a latent variable h would be superfluous – X can be completely modelled with a mean and a covariance. We need to impose some more specific structure on one of these two parameters. A simple additional assumption regards the structure of the error covariance Σ

• $\Sigma = I$! This assumption leads to the probabilistic model of PCA (Principal Component Analysis)

scikitlearn user guide Release 0213

• $\Sigma = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_p^2)$ This model is called FactorAnalysis a classical statistical model The matrix W is sometimes called the “factor loading matrix”

Both models essentially estimate a Gaussian with a lowrank covariance matrix Because both models are probabilistic they can be integrated in more complex models eg Mixture of Factor Analysers One gets very different models eg FastICA if nonGaussian priors on the latent variables are assumed

Factor analysis can produce similar components the columns of its loading matrix to PCA (Principal Component Analysis) However one can not make any general statements about these components eg whether they are orthogonal The main advantage for Factor Analysis over PCA (Principal Component Analysis) is that it can model the variance in every direction of the input space independently heteroscedastic noise

This allows better model selection than probabilistic PCA (Principal Component Analysis) in the presence of heteroscedastic noise Examples

• Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA

Independent component analysis ICA

Independent component analysis separates a multivariate signal into additive subcomponents that are maximally independent It is implemented in scikitlearn using the Fast ICA algorithm Typically ICA is not used for reducing dimensionality but for separating superimposed signals Since the ICA model does not include a noise term for the model to be correct whitening must be applied This can be done internally using the `whiten` argument or manually using one of the PCA (Principal Component Analysis) variants

It is classically used to separate mixed signals a problem known as blind source separation as in the example below ICA can also be used as yet another non linear decomposition that finds components with some sparsity

scikitlearn user guide Release 0213

Examples

- Blind source separation using FastICA
- FastICA on 2D point clouds
- Faces dataset decompositions

Nonnegative matrix factorization NMF (Non-Negative Matrix Factorization) or NNMF

NMF (Non-Negative Matrix Factorization) with the Frobenius norm

NMF is an alternative approach to decomposition that assumes that the data and the components are nonnegative

NMF (Non-Negative Matrix Factorization) can be plugged in instead of PCA (Principal Component Analysis) or its variants in the cases where the data values are nonnegative. It finds a decomposition of samples X into two matrices W and H of nonnegative elements by optimizing the distance between X and the matrix product WH . The most widely used distance function is the squared Frobenius norm which is an obvious extension of the Euclidean norm to matrices

$$\|X - WH\|_F^2 = \sum_{i,j} (X_{ij} - \sum_k W_{ik} H_{kj})^2$$

$$\|X - WH\|_F^2 = \sum_{i,j} X_{ij}^2 - 2 \sum_{i,j,k} X_{ij} W_{ik} H_{kj} + \sum_{i,j,k,l} W_{ik} H_{kj} W_{il} H_{lj}$$

$$\|X - WH\|_F^2 = \sum_{i,j} X_{ij}^2 - 2 \sum_{i,j,k} X_{ij} W_{ik} H_{kj} + \sum_{i,j,k,l} W_{ik} H_{kj} W_{il} H_{lj}$$

$$\|X - WH\|_F^2 = \sum_{i,j} X_{ij}^2 - 2 \sum_{i,j,k} X_{ij} W_{ik} H_{kj} + \sum_{i,j,k,l} W_{ik} H_{kj} W_{il} H_{lj}$$

$$\|X - WH\|_F^2 = \sum_{i,j} X_{ij}^2 - 2 \sum_{i,j,k} X_{ij} W_{ik} H_{kj} + \sum_{i,j,k,l} W_{ik} H_{kj} W_{il} H_{lj}$$

Unlike PCA the representation of a vector is obtained in an additive fashion by superimposing the components without subtracting. Such additive models are efficient for representing images and text.

It has been observed in Hoyer 2004 that when carefully constrained NMF (Non-Negative Matrix Factorization) can produce a parts-based representation of the dataset resulting in interpretable models. The following example displays 16 sparse components found by NMF (Non-Negative Matrix Factorization) from the images in the Olivetti faces dataset in comparison with the PCA (Principal Component Analysis) eigenfaces.

1 “Learning the parts of objects by nonnegative matrix factorization” D Lee S Seung 1999

2 “Nonnegative Matrix Factorization with Sparseness Constraints” P Hoyer 2004

32 Unsupervised learning 407

The `init` attribute determines the initialization method applied which has a great impact on the performance of the method. `NMF` implements the method Nonnegative Double Singular Value Decomposition. `NNDSVD` is based on two SVD processes: one approximating the data matrix, the other approximating positive sections of the resulting partial SVD factors, utilizing an algebraic property of unit rank matrices. The basic `NNDSVD` algorithm is better fit for sparse factorization. Its variants `NNDSVDa` in which all zeros are set equal to the mean of all elements of the data, and `NNDSVDar` in which the zeros are set to random perturbations less than the mean of the data divided by 100 are recommended in the dense case.

Note that the Multiplicative Update ‘`mu`’ solver cannot update zeros present in the initialization so it leads to poorer results when used jointly with the basic `NNDSVD` algorithm which introduces a lot of zeros in this case. `NNDSVDa` or `NNDSVDar` should be preferred.

`NMF` (Non-Negative Matrix Factorization) can also be initialized with correctly scaled random nonnegative matrices by setting `init='random'`. An integer seed or a `RandomState` can also be passed to `random_state` to control reproducibility.

In `NMF`, L1 and L2 priors can be added to the loss function in order to regularize the model. The L2 prior uses the Frobenius norm while the L1 prior uses an elementwise L1 norm. As in `ElasticNet`, we control the combination of L1 and L2 with the `l1_ratio` parameter and the intensity of the regularization with the `alpha` parameter.

Then the priors terms are

$$\frac{\alpha}{2} \|W\|_1 + \frac{\beta}{2} \|H\|_1 - \langle W, H \rangle$$

$$\frac{\alpha}{2} \|W\|_2^2$$

$$\frac{\beta}{2} \|H\|_2^2$$

$$\frac{\alpha}{2} \|W\|_2^2$$

$$\frac{\beta}{2} \|H\|_2^2$$

4 “SVD based initialization: A head start for nonnegative matrix factorization” C. Boutsidis, E. Gallopoulos, 2008

scikitlearn user guide Release 0213
and the regularized objective function is

$$\frac{1}{2} \|W - WH\|_F^2 + \frac{\lambda}{2} \|W\|_F^2 + \frac{\lambda}{2} \|H\|_F^2$$

NMF (Non-Negative Matrix Factorization) regularizes both W and H The public function nonnegativefactorization allows a finer control through the regularization attribute and may regularize only W only H or both

NMF (Non-Negative Matrix Factorization) with a betadivergence

As described previously the most widely used distance function is the squared Frobenius norm which is an obvious extension of the Euclidean norm to matrices

$$\frac{1}{2} \|W - WH\|_F^2 + \frac{\lambda}{2} \|W\|_F^2 + \frac{\lambda}{2} \|H\|_F^2$$

Other distance functions can be used in NMF (Non-Negative Matrix Factorization) as for example the generalized KullbackLeibler KL divergence also referred as Idivergence

$$\sum_i \left(\frac{W_{ij}}{WH_{ij}} \log \frac{W_{ij}}{WH_{ij}} \right)$$

Or the ItakuraSaito IS divergence

$$\sum_i \left(\frac{W_{ij}}{WH_{ij}} - \log \frac{W_{ij}}{WH_{ij}} - 1 \right)$$

These three distances are special cases of the betadivergence family with $\beta = 2, 1, 0$ respectively The beta divergence are defined by

$$\sum_i \left(\frac{W_{ij}}{WH_{ij}} - 1 + \frac{1}{\beta} \left(\frac{W_{ij}}{WH_{ij}} - 1 \right)^\beta \right)$$

Note that this definition is not valid if $\beta \in [0, 1]$ yet it can be continuously extended to the definitions of $\beta = 0$ and $\beta = 1$ respectively

NMF (Non-Negative Matrix Factorization) implements two solvers using Coordinate Descent ‘cd’ and Multiplicative Update ‘mu’ The ‘mu’ solver can optimize every betadivergence including of course the Frobenius norm $\beta = 2$ the generalized Kullback Leibler divergence $\beta = 1$ and the ItakuraSaito divergence $\beta = 0$ Note that for $\beta \in [1, 2]$ the ‘mu’ solver is significantly faster than for other values of β Note also that with a negative or 0 ie ‘itakurasaito’ the input matrix cannot contain zero values

The ‘cd’ solver can only optimize the Frobenius norm Due to the underlying nonconvexity of NMF (Non-Negative Matrix Factorization) the different solvers may converge to different minima even when optimizing the same distance function

NMF (Non-Negative Matrix Factorization) is best used with the fittransform method which returns the matrix W The matrix H is stored into the fitted model in the components attribute the method transform will decompose a new matrix Xnew based on these stored components

```
import numpy as np
X = np.array([1, 2, 1, 3, 12, 4, 1, 5, 08, 6, 1])
from sklearn.decomposition import NMF (Non-Negative Matrix Factorization)
model = NMF(n_components=2, init='random', random_state=0)
```

6 “Algorithms for nonnegative matrix factorization with the betadivergence” C Fevotte J Idier 2011
5 “Fast local algorithms for large scale nonnegative matrix and tensor factorizations” A Cichocki A Phan 2009
32 Unsupervised learning 409

scikitlearn user guide Release 0213

W modelfittransformX

H modelcomponents

Xnew nparray1 0 1 61 1 0 1 4 32 1 0 4

Wnew modeltransformXnew

Examples

- Faces dataset decompositions
- Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation
- Betadivergence loss functions

References

Latent Dirichlet Allocation LDA

Latent Dirichlet Allocation is a generative probabilistic model for collections of discrete dataset such as text corpora

It is also a topic model that is used for discovering abstract topics from a collection of documents

The graphical model of LDA is a threelevel generative model

410 Chapter 3 User Guide

Note on notations presented in the graphical model above which can be found in Hoffman et al 2013

- The corpus is a collection of D documents
- A document is a sequence of N words
- There are K topics in the corpus
- The boxes represent repeated sampling

In the graphical model each node is a random variable and has a role in the generative process A shaded node indicates an observed variable and an unshaded node indicates a hidden latent variable In this case words in the corpus are the only data that we observe The latent variables determine the random mixture of topics in the corpus and the distribution of words in the documents The goal of LDA is to use the observed words to infer the hidden topic structure

When modeling text corpora the model assumes the following generative process for a corpus with D documents and K topics with V corresponding to n components in the API (Application Programming Interface)

- 1 For each topic $k \in K$ draw $\theta_k \sim \text{Dirichlet}(\alpha)$ This provides a distribution over the words ie the probability of a word appearing in topic k corresponds to θ_k
- 2 For each document $d \in D$ draw the topic proportions $\phi_d \sim \text{Dirichlet}(\beta)$ corresponds to ϕ_d

3 For each word n in document d

- 1 Draw the topic assignment $z_n \sim \text{Multinomial}(\phi_d)$
- 2 Draw the observed word $w_n \sim \text{Multinomial}(\theta_{z_n})$

For parameter estimation the posterior distribution is

$$p(\theta, \phi, z | w) \propto \prod_{k=1}^K \Gamma(\alpha) \prod_{v=1}^V \theta_{kv}^{\alpha_{kv} - 1} \prod_{d=1}^D \Gamma(\beta) \prod_{v=1}^V \phi_{dv}^{\beta_{dv} - 1} \prod_{n=1}^N \Gamma(\theta_{z_n, w_n}) \theta_{z_n, w_n}$$

Since the posterior is intractable variational Bayesian method uses a simpler distribution $q(\theta, \phi, z)$ to approximate it and those variational parameters are optimized to maximize the Evidence Lower Bound ELBO

$$\log p(w) \geq \mathbb{E}_{q(\theta, \phi, z)} [\log p(w | \theta, \phi, z)] - \text{KL}(q(\theta, \phi, z) || p(\theta, \phi, z))$$

Maximizing ELBO is equivalent to minimizing the KullbackLeiblerKL divergence between $q(\theta, \phi, z)$ and the true posterior $p(\theta, \phi, z | w)$

LatentDirichletAllocation implements the online variational Bayes algorithm and supports both online and batch update methods While the batch method updates variational variables after each full pass through the data the online method updates variational variables from minibatch data points

Note Although the online method is guaranteed to converge to a local optimum point the quality of the optimum point and the speed of convergence may depend on minibatch size and attributes related to learning rate setting

scikitlearn user guide Release 0213

WhenLatentDirichletAllocation is applied on a “documentterm” matrix the matrix will be decomposed into a “topicterm” matrix and a “documenttopic” matrix While “topicterm” matrix is stored as components in the model “documenttopic” matrix can be calculated from transform method LatentDirichletAllocation also implements partialfit method This is used when data can be fetched sequentially

Examples

- Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation

References

- “Latent Dirichlet Allocation” D Blei A Ng M Jordan 2003
- “Online Learning for Latent Dirichlet Allocation” M Hoffman D Blei F Bach 2010
- “Stochastic Variational Inference” M Hoffman D Blei C Wang J Paisley 2013

See also Dimensionality reduction for dimensionality reduction with Neighborhood Components Analysis

326 Covariance estimation

Many statistical problems require the estimation of a population’s covariance matrix which can be seen as an estimation of data set scatter plot shape Most of the time such an estimation has to be done on a sample whose properties size structure homogeneity have a large influence on the estimation’s quality The sklearncovariance package provides tools for accurately estimating a population’s covariance matrix under various settings We assume that the observations are independent and identically distributed iid

Empirical covariance

The covariance matrix of a data set is known to be well approximated by the classical maximum likelihood estimator or “empirical covariance” provided the number of observations is large enough compared to the number of features the variables describing the observations More precisely the Maximum Likelihood Estimator of a sample is an unbiased estimator of the corresponding population’s covariance matrix

The empirical covariance matrix of a sample can be computed using the empiricalcovariance function of the package or by fitting an EmpiricalCovariance object to the data sample with the EmpiricalCovariance fit method Be careful that results depend on whether the data are centered so one may want to use the assumecentered parameter accurately More precisely if assumecenteredFalse then the test set is supposed to have the same mean vector as the training set If not both should be centered by the user and assumecenteredTrue should be used

Examples

- See Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood for an example on how to fit an EmpiricalCovariance object to data

Shrunk Covariance

Basic shrinkage

Despite being an unbiased estimator of the covariance matrix the Maximum Likelihood Estimator is not a good estimator of the eigenvalues of the covariance matrix so the precision matrix obtained from its inversion is not accurate. Sometimes it even occurs that the empirical covariance matrix cannot be inverted for numerical reasons. To avoid such an inversion problem a transformation of the empirical covariance matrix has been introduced: the shrinkage. In scikitlearn this transformation with a userdefined shrinkage coefficient can be directly applied to a precomputed covariance with the `shrunkcovariance` method. Also a shrunk estimator of the covariance can be fitted to data with a `ShrunkCovariance` object and its `ShrunkCovariancefit` method. Again results depend on whether the data are centered so one may want to use the `assume_centered` parameter accurately.

Mathematically this shrinkage consists in reducing the ratio between the smallest and the largest eigenvalues of the empirical covariance matrix. It can be done by simply shifting every eigenvalue according to a given offset which is equivalent of finding the L_2 penalized Maximum Likelihood Estimator of the covariance matrix. In practice shrinkage boils down to a simple convex transformation $\hat{\Sigma}_{shrunk} = (1 - \alpha) \hat{\Sigma} + \alpha \text{Tr}(\hat{\Sigma}) / d$

Choosing the amount of shrinkage amounts to setting a biasvariance tradeoff and is discussed below.

- Examples
- See Shrinkage covariance estimation: LedoitWolf vs OAS and maxlikelihood for an example on how to fit a `ShrunkCovariance` object to data.

LedoitWolf shrinkage

In their 2004 paper¹ O Ledoit and M Wolf propose a formula to compute the optimal shrinkage coefficient α that minimizes the Mean Squared Error between the estimated and the real covariance matrix. The `LedoitWolf` estimator of the covariance matrix can be computed on a sample with the `ledoitwolf` function of the `sklearn.covariance` package or it can be otherwise obtained by fitting a `LedoitWolf` object to the same sample.

Note: Case when population covariance matrix is isotropic

It is important to note that when the number of samples is much larger than the number of features one would expect that no shrinkage would be necessary. The intuition behind this is that if the population covariance is full rank when the number of samples grows the sample covariance will also become positive definite. As a result no shrinkage would be necessary and the method should automatically do this.

This however is not the case in the `LedoitWolf` procedure when the population covariance happens to be a multiple of the identity matrix. In this case the `LedoitWolf` shrinkage estimate approaches 1 as the number of samples increases. This indicates that the optimal estimate of the covariance matrix in the `LedoitWolf` sense is multiple of the identity. Since the population covariance is already a multiple of the identity matrix the `LedoitWolf` solution is indeed a reasonable estimate.

¹O Ledoit and M Wolf “A WellConditioned Estimator for LargeDimensional Covariance Matrices” Journal of Multivariate Analysis V ol ume 88 Issue 2 February 2004 pages 365411
32 Unsupervised learning 413

scikitlearn user guide Release 0213

Examples

- See Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood for an example on how to fit a LedoitWolf object to data and for visualizing the performances of the LedoitWolf estimator in terms of likelihood

References

Oracle Approximating Shrinkage

Under the assumption that the data are Gaussian distributed Chen et al² derived a formula aimed at choosing a shrinkage coefficient that yields a smaller Mean Squared Error than the one given by Ledoit and Wolf’s formula The resulting estimator is known as the Oracle Shrinkage Approximating estimator of the covariance

The OAS estimator of the covariance matrix can be computed on a sample with the oas function of the sklearn covariance package or it can be otherwise obtained by fitting an OAS object to the same sample

Fig 37 Biasvariance tradeoff when setting the shrinkage comparing the choices of LedoitWolf and OAS estimators

References

Examples

- See Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood for an example on how to fit an OAS object to data

²Chen et al “Shrinkage Algorithms for MMSE Covariance Estimation” IEEE Trans on Sign Proc Volume 58 Issue 10 October 2010
414 Chapter 3 User Guide

scikitlearn user guide Release 0213

- See LedoitWolf vs OAS estimation to visualize the Mean Squared Error difference between a LedoitWolf and anOAS estimator of the covariance

Sparse inverse covariance

The matrix inverse of the covariance matrix often called the precision matrix is proportional to the partial correlation matrix It gives the partial independence relationship In other words if two features are independent conditionally on the others the corresponding coefficient in the precision matrix will be zero This is why it makes sense to estimate a sparse precision matrix the estimation of the covariance matrix is better conditioned by learning independence relations from the data This is known as covariance selection

In the smallsamples situation in which nsamples is on the order of nfeatures or smaller sparse inverse covariance estimators tend to work better than shrunk covariance estimators However in the opposite situation or for very correlated data they can be numerically unstable In addition unlike shrinkage estimators sparse estimators are able to recover offdiagonal structure

TheGraphicalLasso estimator uses an l1 penalty to enforce sparsity on the precision matrix the higher its alpha parameter the more sparse the precision matrix The corresponding GraphicalLassoCV object uses crossvalidation to automatically set the alpha parameter

Note Structure recovery

Recovering a graphical structure from correlations in the data is a challenging thing If you are interested in such recovery keep in mind that

- Recovery is easier from a correlation matrix than a covariance matrix standardize your observations before runningGraphicalLasso
- If the underlying graph has nodes with much more connections than the average node the algorithm will miss some of these connections

scikitlearn user guide Release 0213

Fig 38 A comparison of maximum likelihood shrinkage and sparse estimates of the covariance and precision matrix in the very small samples settings

- If your number of observations is not large compared to the number of edges in your underlying graph you will not recover it
- Even if you are in favorable recovery conditions the alpha parameter chosen by crossvalidation eg using the GraphicalLassoCV object will lead to selecting too many edges However the relevant edges will have heavier weights than the irrelevant ones

The mathematical formulation is the following

$$\hat{\Sigma} = \operatorname{argmin}_{\Sigma} \left\{ \operatorname{tr}(\Sigma S) - \log \det \Sigma + \lambda \|\Sigma\|_1 \right\}$$

Where Σ is the precision matrix to be estimated and S is the sample covariance matrix $\|\Sigma\|_1$ is the sum of the absolute values of offdiagonal coefficients of Σ The algorithm employed to solve this problem is the GLasso algorithm from the Friedman 2008 Biostatistics paper It is the same algorithm as in the R glasso package

Examples

- Sparse inverse covariance estimation example on synthetic data showing some recovery of a structure and comparing to other covariance estimators
- Visualizing the stock market structure example on real stock market data finding which symbols are most linked

References

- Friedman et al “Sparse inverse covariance estimation with the graphical lasso” Biostatistics 9 pp 432 2008

scikitlearn user guide Release 0213

Robust Covariance Estimation

Real data sets are often subject to measurement or recording errors Regular but uncommon observations may also appear for a variety of reasons Observations which are very uncommon are called outliers The empirical covariance estimator and the shrunk covariance estimators presented above are very sensitive to the presence of outliers in the data Therefore one should use robust covariance estimators to estimate the covariance of its real data sets Alternatively robust covariance estimators can be used to perform outlier detection and discarddownweight some observations according to further processing of the data

Thesklearncovariance package implements a robust estimator of covariance the Minimum Covariance De terminant3

Minimum Covariance Determinant

The Minimum Covariance Determinant estimator is a robust estimator of a data set’s covariance introduced by PJ Rousseeuw in3 The idea is to find a given proportion h of “good” observations which are not outliers and compute their empirical covariance matrix This empirical covariance matrix is then rescaled to compensate the performed selection of observations “consistency step” Having computed the Minimum Covariance Determinant estimator one can give weights to observations according to their Mahalanobis distance leading to a reweighted estimate of the covariance matrix of the data set “reweighting step”

Rousseeuw and Van Driessen4developed the FastMCD algorithm in order to compute the Minimum Covariance Determinant This algorithm is used in scikitlearn when fitting an MCD object to data The FastMCD algorithm also computes a robust estimate of the data set location at the same time

Raw estimates can be accessed as rawlocation andrawcovariance attributes of a MinCovDet robust covariance estimator object

References

Examples

- See Robust vs Empirical covariance estimate for an example on how to fit a MinCovDet object to data and see how the estimate remains accurate despite the presence of outliers
- See Robust covariance estimation and Mahalanobis distances relevance to visualize the difference between EmpiricalCovariance andMinCovDet covariance estimators in terms of Mahalanobis distance so we get a better estimate of the precision matrix too

3P J Rousseeuw Least median of squares regression J Am Stat Ass 79871 1984

4A Fast Algorithm for the Minimum Covariance Determinant Estimator 1999 American Statistical Association and the American Society for Quality TECHNOMETRICS

32 Unsupervised learning 417

scikitlearn user guide Release 0213

Influence of outliers on location and covariance

estimatesSeparating inliers from outliers using a Mahalanobis distance

327 Novelty and Outlier Detection

Many applications require being able to decide whether a new observation belongs to the same distribution as existing observations it is an inlier or should be considered as different it is an outlier Often this ability is used to clean real data sets Two important distinctions must be made

outlier detection The training data contains outliers which are defined as observations that are far from the others Outlier detection estimators thus try to fit the regions where the training data is the most concentrated ignoring the deviant observations

novelty detection The training data is not polluted by outliers and we are interested in detecting whether anew observation is an outlier In this context an outlier is also called a novelty

Outlier detection and novelty detection are both used for anomaly detection where one is interested in detecting abnormal or unusual observations Outlier detection is then also known as unsupervised anomaly detection and novelty detection as semisupervised anomaly detection In the context of outlier detection the outliersanomalies cannot form a dense cluster as available estimators assume that the outliersanomalies are located in low density regions On the contrary in the context of novelty detection noveltiesanomalies can form a dense cluster as long as they are in a low density region of the training data considered as normal in this context

The scikitlearn project provides a set of machine learning tools that can be used both for novelty or outlier detection This strategy is implemented with objects learning in an unsupervised way from the data

estimatorfitXtrain

new observations can then be sorted as inliers or outliers with a predict method

estimatorpredictXtest

Inliers are labeled 1 while outliers are labeled 1 The predict method makes use of a threshold on the raw scoring function computed by the estimator This scoring function is accessible through the scoresamples method while the threshold can be controlled by the contamination parameter

Thedecisionfunction method is also defined from the scoring function in such a way that negative values are outliers and nonnegative ones are inliers

estimatordecisionfunctionXtest

418 Chapter 3 User Guide

scikitlearn user guide Release 0213

Note that neighborsLocalOutlierFactor does not support predict decisionfunction and scoresamples methods by default but only a fitpredict method as this estimator was originally meant to be applied for outlier detection The scores of abnormality of the training samples are accessible through the negativeoutlierfactor attribute
If you really want to use neighborsLocalOutlierFactor for novelty detection ie predict labels or compute the score of abnormality of new unseen data you can instantiate the estimator with the novelty parameter set to True before fitting the estimator In this case fitpredict is not available

Warning Novelty detection with Local Outlier Factor

When novelty is set to True be aware that you must only use predict decisionfunction and scoresamples on new unseen data and not on the training samples as this would lead to wrong results The scores of abnormality of the training samples are always accessible through the negativeoutlierfactor attribute

The behavior of neighborsLocalOutlierFactor is summarized in the following table

| Method | Outlier detection | Novelty detection |
|------------------|---------------------------|----------------------|
| fitpredict | OK | Not available |
| predict | Not available | Use only on new data |
| decisionfunction | Not available | Use only on new data |
| scoresamples | Use negativeoutlierfactor | Use only on new data |

Overview of outlier detection methods

A comparison of the outlier detection algorithms in scikitlearn Local Outlier Factor LOF does not show a decision boundary in black as it has no predict method to be applied on new data when it is used for outlier detection ensembleIsolationForest and neighborsLocalOutlierFactor perform reasonably well on the data sets considered here The svmOneClassSVM is known to be sensitive to outliers and thus does not perform very well for outlier detection Finally covarianceEllipticEnvelope assumes the data is Gaussian and learns an ellipse For more details on the different estimators refer to the example Comparing anomaly detection algorithms for outlier detection on toy datasets and the sections hereunder

Examples

- See Comparing anomaly detection algorithms for outlier detection on toy datasets for a comparison of the svmOneClassSVM the ensembleIsolationForest the neighborsLocalOutlierFactor and covarianceEllipticEnvelope

Novelty Detection

Consider a data set of n observations from the same distribution described by d features Consider now that we add one more observation to that data set Is the new observation so different from the others that we can doubt it is regular ie does it come from the same distribution Or on the contrary is it so similar to the other that we cannot distinguish it from the original observations This is the question addressed by the novelty detection tools and methods In general it is about to learn a rough close frontier delimiting the contour of the initial observations distribution plotted in embedding d -dimensional space Then if further observations lay within the frontier delimited subspace

scikitlearn user guide Release 0213

they are considered as coming from the same population than the initial observations Otherwise if they lay outside the frontier we can say that they are abnormal with a given confidence in our assessment

The OneClass SVM (Support Vector Machine) has been introduced by Schölkopf et al for that purpose and implemented in the Support Vector Machines module in the svmOneClassSVM object It requires the choice of a kernel and a scalar parameter to define a frontier The RBF (Radial Basis Function) kernel is usually chosen although there exists no exact formula or algorithm to set its bandwidth parameter This is the default in the scikitlearn implementation The γ parameter also known as the margin of the OneClass SVM (Support Vector Machine) corresponds to the probability of finding a new but regular observation outside the frontier

References

- Estimating the support of a highdimensional distribution Schölkopf Bernhard et al Neural computation 137 2001 14431471

Examples

- See Oneclass SVM (Support Vector Machine) with nonlinear kernel RBF (Radial Basis Function) for visualizing the frontier learned around some data
- svmOneClassSVM object
- Species distribution modeling

Outlier Detection

Outlier detection is similar to novelty detection in the sense that the goal is to separate a core of regular observations from some polluting ones called outliers Yet in the case of outlier detection we don't have a clean data set representing the population of regular observations that can be used to train any tool

scikitlearn user guide Release 0213

Fitting an elliptic envelope

One common way of performing outlier detection is to assume that the regular data come from a known distribution eg data are Gaussian distributed From this assumption we generally try to define the “shape” of the data and can define outlying observations as observations which stand far enough from the fit shape The scikitlearn provides an object covarianceEllipticEnvelope that fits a robust covariance estimate to the data and thus fits an ellipse to the central data points ignoring points outside the central mode For instance assuming that the inlier data are Gaussian distributed it will estimate the inlier location and covariance in a robust way ie without being influenced by outliers The Mahalanobis distances obtained from this estimate is used to derive a measure of outlyingness This strategy is illustrated below

Examples

- See Robust covariance estimation and Mahalanobis distances relevance for an illustration of the difference between using a standard covarianceEmpiricalCovariance or a robust estimate covarianceMinCovDet of location and covariance to assess the degree of outlyingness of an observation

References

- Rousseeuw PJ Van Driessen K “A fast algorithm for the minimum covariance determinant estimator” Technometrics 41:3 212 1999
- 422 Chapter 3 User Guide

Isolation Forest

One efficient way of performing outlier detection in highdimensional datasets is to use random forests The ensembleIsolationForest ‘isolates’ observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature Since recursive partitioning can be represented by a tree structure the number of splittings required to isolate a sample is equivalent to the path length from the root node to the terminating node This path length averaged over a forest of such random trees is a measure of normality and our decision function Random partitioning produces noticeably shorter paths for anomalies Hence when a forest of random trees collectively produce shorter path lengths for particular samples they are highly likely to be anomalies The implementation of ensembleIsolationForest is based on an ensemble of tree ExtraTreeRegressor Following Isolation Forest original paper the maximum depth of each tree is set to $\lceil \log_2 n \rceil$ where n is the number of samples used to build the tree see Liu et al 2008 for more details This algorithm is illustrated below The ensembleIsolationForest supportswarmstartTrue which allows you to add more trees to an already fitted model

```
from sklearnensemble import IsolationForest
import numpy as np
X = np.array([1, 2, 1, 3, 2, 0, 0, 20, 50, 3, 5])
clf = IsolationForest(n_estimators=10, warmstart=True)
clf.fit(X) # fit 10 trees
clf.set_params(n_estimators=20) # add 10 more trees
clf.fit(X) # fit the added trees
```

Examples

- See IsolationForest example for an illustration of the use of IsolationForest

32 Unsupervised learning 423

scikitlearn user guide Release 0213

- See Comparing anomaly detection algorithms for outlier detection on toy datasets for a comparison of ensembleIsolationForest withneighborsLocalOutlierFactor svmOneClassSVM tuned to perform like an outlier detection method and a covariancebased outlier detection with covarianceEllipticEnvelope

References

- Liu Fei Tony Ting Kai Ming and Zhou ZhiHua “Isolation forest” Data Mining 2008 ICDM’08 Eighth IEEE International Conference on

Local Outlier Factor

Another efficient way to perform outlier detection on moderately high dimensional datasets is to use the Local Outlier Factor LOF algorithm

TheneighborsLocalOutlierFactor LOF algorithm computes a score called local outlier factor reflecting the degree of abnormality of the observations. It measures the local density deviation of a given data point with respect to its neighbors. The idea is to detect the samples that have a substantially lower density than their neighbors. In practice, the local density is obtained from the k nearest neighbors. The LOF score of an observation is equal to the ratio of the average local density of its k nearest neighbors and its own local density. A normal instance is expected to have a local density similar to that of its neighbors, while abnormal data are expected to have much smaller local density.

The number k of neighbors considered (alias parameter nneighbors) is typically chosen 1 greater than the minimum number of objects a cluster has to contain so that other objects can be local outliers relative to this cluster and 2 smaller than the maximum number of close-by objects that can potentially be local outliers. In practice, such information is generally not available, and taking nneighbors=20 appears to work well in general. When the proportion of outliers is high (i.e. greater than 10%) as in the example below, nneighbors should be greater (nneighbors=35 in the example below).

The strength of the LOF algorithm is that it takes both local and global properties of datasets into consideration; it can perform well even in datasets where abnormal samples have different underlying densities. The question is not how isolated the sample is, but how isolated it is with respect to the surrounding neighborhood.

When applying LOF for outlier detection, there are no predict, decisionfunction, and scoresamples methods, but only a fitpredict method. The scores of abnormality of the training samples are accessible through the negativeoutlierfactor attribute. Note that predict, decisionfunction, and scoresamples can be used on new, unseen data when LOF is applied for novelty detection (i.e. when the novelty parameter is set to True). See Novelty detection with Local Outlier Factor.

This strategy is illustrated below.

Examples

- See Outlier detection with Local Outlier Factor LOF for an illustration of the use of neighbors

LocalOutlierFactor

- See Comparing anomaly detection algorithms for outlier detection on toy datasets for a comparison with other anomaly detection methods

scikitlearn user guide Release 0213

References

- Breunig Kriegel Ng and Sander 2000 LOF identifying densitybased local outliers Proc ACM SIGMOD

Novelty detection with Local Outlier Factor

To use neighborsLocalOutlierFactor for novelty detection ie predict labels or compute the score of abnormality of new unseen data you need to instantiate the estimator with the novelty parameter set to True before fitting the estimator

lof = LocalOutlierFactornovelty=True

lof.fit(Xtrain)

Note that fit_predict is not available in this case

Warning Novelty detection with Local Outlier Factor'

When novelty is set to True be aware that you must only use predict decision function

and score samples on new unseen data and not on the training samples as this would lead to

wrong results The scores of abnormality of the training samples are always accessible through the

negative_outlier_factor_ attribute

Novelty detection with Local Outlier Factor is illustrated below

328 Density Estimation

Density estimation walks the line between unsupervised learning feature engineering and data modeling Some of the most popular and useful density estimation techniques are mixture models such as Gaussian Mixtures `sklearn.mixture.GaussianMixture` and neighborbased approaches such as the kernel density estimate `sklearn.neighbors.KernelDensity` Gaussian Mixtures are discussed more fully in the context of clustering because the technique is also useful as an unsupervised clustering scheme

Density estimation is a very simple concept and most people are already familiar with one common density estimation technique the histogram

Density Estimation Histograms

A histogram is a simple visualization of data where bins are defined and the number of data points within each bin is tallied An example of a histogram can be seen in the upperleft panel of the following figure

scikitlearn user guide Release 0213

A major problem with histograms however is that the choice of binning can have a disproportionate effect on the resulting visualization Consider the upperright panel of the above figure It shows a histogram over the same data with the bins shifted right The results of the two visualizations look entirely different and might lead to different interpretations of the data

Intuitively one can also think of a histogram as a stack of blocks one block per point By stacking the blocks in the appropriate grid space we recover the histogram But what if instead of stacking the blocks on a regular grid we center each block on the point it represents and sum the total height at each location This idea leads to the lowerleft visualization It is perhaps not as clean as a histogram but the fact that the data drive the block locations mean that it is a much better representation of the underlying data

This visualization is an example of a kernel density estimation in this case with a tophat kernel ie a square block at each point We can recover a smoother distribution by using a smoother kernel The bottomright plot shows a Gaussian kernel density estimate in which each point contributes a Gaussian curve to the total The result is a smooth density estimate which is derived from the data and functions as a powerful nonparametric model of the distribution of points

Kernel Density Estimation

Kernel density estimation in scikitlearn is implemented in the sklearnneighborsKernelDensity estimator which uses the Ball Tree or KD Tree for efficient queries see Nearest Neighbors for a discussion of these Though the above example uses a 1D data set for simplicity kernel density estimation can be performed in any number of dimensions though in practice the curse of dimensionality causes its performance to degrade in high dimensions In the following figure 100 points are drawn from a bimodal distribution and the kernel density estimates are shown for three choices of kernels

scikitlearn user guide Release 0213

It's clear how the kernel shape affects the smoothness of the resulting distribution The scikitlearn kernel density estimator can be used as follows

```
from sklearn.neighbors.kde import KernelDensity
import numpy as np
X = np.array([1, 2, 1, 3, 2, 1, 1, 2, 1, 3, 2])
kde = KernelDensity(kernel='gaussian', bandwidth=0.2).fit(X)
kde.score_samples(X)
array([0.41075698, 0.41075698, 0.41076071, 0.41075698, 0.41075698,
       0.41076071])
```

Here we have used kernelgaussian as seen above Mathematically a kernel is a positive function $K(x, x')$ which is controlled by the bandwidth parameter h Given this kernel form the density estimate at a point x within a group of points x_1, \dots, x_n is given by

$$\hat{p}(x) = \frac{1}{n} \sum_{i=1}^n K(x, x_i) / h$$

The bandwidth here acts as a smoothing parameter controlling the tradeoff between bias and variance in the result A large bandwidth leads to a very smooth ie highbias density distribution A small bandwidth leads to an unsmooth ie highvariance density distribution

sklearn.neighbors.KernelDensity implements several common kernel forms which are shown in the following figure

scikitlearn user guide Release 0213

The form of these kernels is as follows

- Gaussian kernel kernel gaussian

$$K(x,y)=\exp\left(-\frac{\|x-y\|^2}{2h^2}\right)$$

- Tophat kernel kernel tophat

$$K(x,y)=\frac{1}{h}\exp\left(-\frac{\|x-y\|}{h}\right)$$

- Epanechnikov kernel kernel epanechnikov

$$K(x,y)=\frac{3}{4h}\left(1-\frac{\|x-y\|^2}{h^2}\right)$$

- Exponential kernel kernel exponential

$$K(x,y)=\exp\left(-\frac{\|x-y\|}{h}\right)$$

- Linear kernel kernel linear

$$K(x,y)=\frac{1}{h}\left(1-\frac{\|x-y\|}{h}\right)$$

- Cosine kernel kernel cosine

$$K(x,y)=\cos\left(\frac{\|x-y\|}{h}\right)$$

$$K(x,y)=\frac{1}{2h}\exp\left(-\frac{\|x-y\|}{h}\right)$$

The kernel density estimator can be used with any of the valid distance metrics see sklearnneighbors

DistanceMetric for a list of available metrics though the results are properly normalized only for the Euclidean

metric One particularly useful metric is the Haversine distance which measures the angular distance between points

on a sphere Here is an example of using a kernel density estimate for a visualization of geospatial data in this case

the distribution of observations of two different species on the South American continent

One other useful application of kernel density estimation is to learn a nonparametric generative model of a dataset in order to efficiently draw new samples from this generative model Here is an example of using this process to create a new set of handwritten digits using a Gaussian kernel learned on a PCA (Principal Component Analysis) projection of the data The “new” data consists of linear combinations of the input data with weights probabilistically drawn given the KDE model

scikitlearn user guide Release 0213

Examples

- Simple 1D Kernel Density Estimation computation of simple kernel density estimates in one dimension
- Kernel Density Estimation an example of using Kernel Density estimation to learn a generative model of the handwritten digits data and drawing new samples from this model
- Kernel Density Estimate of Species Distributions an example of Kernel Density estimation using the Haver sine distance metric to visualize geospatial data

329 Neural network models unsupervised

Restricted Boltzmann machines

Restricted Boltzmann machines RBM are unsupervised nonlinear feature learners based on a probabilistic model

The features extracted by an RBM or a hierarchy of RBMs often give good results when fed into a linear classifier

such as a linear SVM (Support Vector Machine) or a perceptron

The model makes assumptions regarding the distribution of inputs At the moment scikitlearn only provides

BernoulliRBM which assumes the inputs are either binary values or values between 0 and 1 each encoding the

probability that the specific feature would be turned on

The RBM tries to maximize the likelihood of the data using a particular graphical model The parameter learning

algorithm used Stochastic Maximum Likelihood prevents the representations from straying far from the input data

which makes them capture interesting regularities but makes the model less useful for small datasets and usually not

useful for density estimation

The method gained popularity for initializing deep neural networks with the weights of independent RBMs This

method is known as unsupervised pretraining

Examples

- Restricted Boltzmann Machine features for digit classification

Graphical model and parametrization

The graphical model of an RBM is a fullyconnected bipartite graph

32 Unsupervised learning 431

The nodes are random variables whose states depend on the state of the other nodes they are connected to The model is therefore parameterized by the weights of the connections as well as one intercept bias term for each visible and hidden unit omitted from the image for simplicity

The energy function measures the quality of a joint assignment

$$E(v,h) = -\sum_{i,j} w_{ij} v_i h_j - \sum_i b_i v_i - \sum_j c_j h_j$$

In the formula above b and c are the intercept vectors for the visible and hidden layers respectively The joint

probability of the model is defined in terms of the energy

$$E(\mathbf{v}, \mathbf{h}) = -\sum_i v_i \sum_j w_{ij} h_j$$

The word restricted refers to the bipartite structure of the model which prohibits direct interaction between hidden units or between visible units This means that the following conditional independencies are assumed

$$h_i \perp h_j \mid \mathbf{v}$$
$$v_i \perp v_j \mid \mathbf{h}$$

The bipartite structure allows for the use of efficient block Gibbs sampling for inference

Bernoulli Restricted Boltzmann machines

In the BernoulliRBM all units are binary stochastic units This means that the input data should either be binary or realvalued between 0 and 1 signifying the probability that the visible unit would turn on or off This is a good model for character recognition where the interest is on which pixels are active and which aren't For images of natural scenes it no longer fits because of background depth and the tendency of neighbouring pixels to take the same values The conditional probability distribution of each unit is given by the logistic sigmoid activation function of the input it receives

$$p(h_i = 1 | \mathbf{v}) = \sigma(\sum_j w_{ij} v_j)$$
$$p(v_i = 1 | \mathbf{h}) = \sigma(\sum_j w_{ji} h_j)$$

where σ is the logistic sigmoid function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Stochastic Maximum Likelihood learning

The training algorithm implemented in BernoulliRBM is known as Stochastic Maximum Likelihood SML or Persistent Contrastive Divergence PCD Optimizing maximum likelihood directly is infeasible because of the form of the data likelihood

$$\log p(\mathbf{v}, \mathbf{h}) = \log \sum_{\mathbf{h}} p(\mathbf{h} | \mathbf{v}) p(\mathbf{v} | \mathbf{h})$$

For simplicity the equation above is written for a single training example The gradient with respect to the weights is formed of two terms corresponding to the ones above They are usually known as the positive gradient and the negative gradient because of their respective signs In this implementation the gradients are estimated over minibatches of samples

In maximizing the loglikelihood the positive gradient makes the model prefer hidden states that are compatible with the observed training data Because of the bipartite structure of RBMs it can be computed efficiently The negative gradient however is intractable Its goal is to lower the energy of joint states that the model prefers therefore making it stay true to the data It can be approximated by Markov chain Monte Carlo using block Gibbs sampling by iteratively sampling each of \mathbf{v} and \mathbf{h} given the other until the chain mixes Samples generated in this way are sometimes referred as fantasy particles This is inefficient and it is difficult to determine whether the Markov chain mixes

The Contrastive Divergence method suggests to stop the chain after a small number of iterations k usually even 1

This method is fast and has low variance but the samples are far from the model distribution

scikitlearn user guide Release 0213

Persistent Contrastive Divergence addresses this. Instead of starting a new chain each time the gradient is needed and performing only one Gibbs sampling step in PCD we keep a number of chains (fantasy particles) that are updated \square Gibbs steps after each weight update. This allows the particles to explore the space more thoroughly.

References

- “A fast learning algorithm for deep belief nets” G Hinton S Osindero Y W Teh 2006
- “Training Restricted Boltzmann Machines using Approximations to the Likelihood Gradient” T Tieleman 2008

33 Model selection and evaluation

331 Crossvalidation evaluating estimator performance

Learning the parameters of a prediction function and testing it on the same data is a methodological mistake: a model that would just repeat the labels of the samples that it has just seen would have a perfect score but would fail to predict anything useful on yet-unseen data. This situation is called overfitting. To avoid it, it is common practice when performing a supervised machine learning experiment to hold out part of the available data as a test set. X_{test} y_{test} . Note that the word “experiment” is not intended to denote academic use only because even in commercial settings machine learning usually starts out experimentally. Here is a flowchart of typical cross validation workflow in model training. The best parameters can be determined by grid search techniques.

In scikitlearn a random split into training and test sets can be quickly computed with the `train_test_split` helper function. Let’s load the iris data set to fit a linear support vector machine on it.

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn import datasets
from sklearn import svm
iris = datasets.load_iris()
iris_data_shape = iris.target_shape
150 4 150
```

We can now quickly sample a training set while holding out 40% of the data for testing, evaluating our classifier.


```
scikitlearn user guide Release 0213
Xtrain Xtest ytrain ytest traintestsplit
irisdata iristarget testsize04 randomstate0
Xtrainshape ytrainshape
90 4 90
Xtestshape ytestshape
60 4 60
clf svmSVCKernellinear C1fitXtrain ytrain
clfscoreXtest ytest
096
```

When evaluating different settings “hyperparameters” for estimators such as the Csetting that must be manually set for an SVM (Support Vector Machine) there is still a risk of overfitting on the test set because the parameters can be tweaked until the estimator performs optimally This way knowledge about the test set can “leak” into the model and evaluation metrics no longer report on generalization performance To solve this problem yet another part of the dataset can be held out as a so called “validation set” training proceeds on the training set after which evaluation is done on the validation set and when the experiment seems to be successful final evaluation can be done on the test set However by partitioning the available data into three sets we drastically reduce the number of samples which can be used for learning the model and the results can depend on a particular random choice for the pair of train validation sets

A solution to this problem is a procedure called crossvalidation CV (Cross-Validation) for short A test set should still be held out for final evaluation but the validation set is no longer needed when doing CV (Cross-Validation) In the basic approach called kfold CV (Cross-Validation) training set is split into ksmaller sets other approaches are described below but generally follow the same principles

The following procedure is followed for each of the k“folds”

- A model is trained using $\frac{n-1}{k}$ of the folds as training data
- the resulting model is validated on the remaining part of the data ie it is used as a test set to compute a performance measure such as accuracy

The performance measure reported by kfold crossvalidation is then the average of the values computed in the loop

This approach can be computationally expensive but does not waste too much data as is the case when fixing an arbitrary validation set which is a major advantage in problems such as inverse inference where the number of samples is very small

scikitlearn user guide Release 0213

Computing crossvalidated metrics

The simplest way to use crossvalidation is to call the crossvalscore helper function on the estimator and the dataset

The following example demonstrates how to estimate the accuracy of a linear kernel support vector machine on the iris dataset by splitting the data fitting a model and computing the score 5 consecutive times with different splits each time

```
from sklearn.model_selection import crossvalscore
clf = svm.SVC(kernel='linear', C=1)
scores = crossvalscore(clf, iris.data, iris.target, cv=5)
scores
array([0.96, 1.   , 0.96, 0.96, 1.   ])
```

The mean score and the 95 confidence interval of the score estimate are hence given by

```
print(Accuracy(0.2f, 0.2f), scores.mean(), scores.std() * 2)
Accuracy: 0.98 ± 0.03
```

By default the score computed at each CV (Cross-Validation) iteration is the score method of the estimator. It is possible to change this by using the scoring parameter

```
from sklearn import metrics
scores = crossvalscore(
    clf, iris.data, iris.target, cv=5, scoring='f1_macro')
scores
array([0.96, 1.   , 0.96, 0.96, 1.   ])
```

See The scoring parameter defining model evaluation rules for details. In the case of the Iris dataset the samples are balanced across target classes hence the accuracy and the F1score are almost equal

When the cv argument is an integer crossvalscore uses the KFold or StratifiedKFold strategies by default the latter being used if the estimator derives from ClassifierMixin

It is also possible to use other cross validation strategies by passing a cross validation iterator instead for instance

```
from sklearn.model_selection import ShuffleSplit
n_samples = iris.data.shape[0]
cv = ShuffleSplit(n_splits=5, test_size=0.3, random_state=0)
crossvalscore(clf, iris.data, iris.target, cv=cv)
array([0.977, 0.977, 1.   , 0.955, 1.   ])
```

Another option is to use an iterable yielding train test splits as arrays of indices for example

```
def customcv2folds(X):
    n = X.shape[0]
    i = 1
    while i < 2:
        idx, npar = generate(i, 1, 2, n, i, 2, dtype=int)
        yield idx, idx
        i = 1
```

```
customcv = customcv2folds(iris.data)
crossvalscore(clf, iris.data, iris.target, cv=customcv)
array([1.   , 0.973])
```

scikitlearn user guide Release 0213

Data transformation with held out data

Just as it is important to test a predictor on data heldout from training preprocessing such as standardization feature selection etc and similar data transformations similarly should be learnt from a training set and applied to heldout data for prediction

```
from sklearn import preprocessing
Xtrain Xtest ytrain ytest traintestsplit
irisdata iristarget testsize04 randomstate0
scaler preprocessingStandardScalerfitXtrain
Xtraintransformed scalertransformXtrain
clf svmSVCC1fitXtraintransformed ytrain
Xtesttransformed scalertransformXtest
clfscoreXtesttransformed ytest
09333
```

APipeline makes it easier to compose estimators providing this behavior under crossvalidation

```
from sklearnpipeline import makepipeline
clf makepipelinepreprocessingStandardScaler svmSVCC1
crossvalscoreclf irisdata iristarget cvcv
```

array0977 0933 0955 0933 0977

SeePipelines and composite estimators

The crossvalidate function and multiple metric evaluation

Thecrossvalidate function differs from crossvalscore in two ways

- It allows specifying multiple metrics for evaluation
- It returns a dict containing fittimes scoretimes and optionally training scores as well as fitted estimators in addition to the test score

For single metric evaluation where the scoring parameter is a string callable or None the keys will be

testscore fittime scoretime

And for multiple metric evaluation the return value is a dict with the following keys

testscorer1name testscorer2name testscorer fittime
scoretime

returntrainscore is set toFalse by default to save computation time To evaluate the scores on the training set as well you need to be set to True

You may also retain the estimator fitted on each training set by setting returnestimatorTrue

The multiple metrics can be specified either as a list tuple or set of predefined scorer names

```
from sklearnmodelselection import crossvalidate
from sklearnmetrics import recallscore
scoring precisionmacro recallmacro
clf svmSVCKernellinear C1 randomstate0
scores crossvaldateclf irisdata iristarget scoringscoring
cv5
sortedscoreskeys
fittime scoretime testprecisionmacro testrecallmacro
scorestestrecallmacro
array096 1 096 096 1
33 Model selection and evaluation 437
```

scikitlearn user guide Release 0213

Or as a dict mapping scorer name to a predefined or custom scoring function

from sklearn.metrics.scorer import makescorer

scoring {'precisionmacro', 'precisionmacro'}

recmacro makescorerrecallscore averagemacro

scores {'crossvalidateclf irisdata iristarget scoringscoring'}

cv5 returntrainscore True

sortedscoreskeys

fittime scoretime testprecmacro testrecmacro

trainprecmacro trainrecmacro

scorestrainrecmacro

array[0.97, 0.97, 0.99, 0.98, 0.98]

Here is an example of crossvalidate using a single metric

scores {'crossvalidateclf irisdata iristarget'}

scoringprecisionmacro cv5

returnestimator True

sortedscoreskeys

estimator fittime scoretime testscore

Obtaining predictions by crossvalidation

The function crossvalpredict has a similar interface to crossvalscore but returns for each element

in the input the prediction that was obtained for that element when it was in the test set Only crossvalidation

strategies that assign all elements to a test set exactly once can be used otherwise an exception is raised

Warning Note on inappropriate usage of crossvalpredict

The result of crossvalpredict may be different from those obtained using crossvalscore as the

elements are grouped in different ways The function crossvalscore takes an average over crossvalidation

folds whereas crossvalpredict simply returns the labels or probabilities from several distinct models

undistinguished Thus crossvalpredict is not an appropriate measure of generalisation error

The function crossvalpredict is appropriate for

- Visualization of predictions obtained from different models
- Model blending When predictions of one supervised estimator are used to train another estimator in ensemble methods

The available cross validation iterators are introduced in the following section

Examples

- Receiver Operating Characteristic ROC with cross validation
 - Recursive feature elimination with crossvalidation
 - Parameter estimation using grid search with crossvalidation
 - Sample pipeline for text feature extraction and evaluation
 - Plotting CrossValidated Predictions
 - Nested versus nonnested crossvalidation
- 438 Chapter 3 User Guide

scikitlearn user guide Release 0213

Cross validation iterators

The following sections list utilities to generate indices that can be used to generate dataset splits according to different cross validation strategies

Crossvalidation iterators for iid data

Assuming that some data is Independent and Identically Distributed iid is making the assumption that all samples stem from the same generative process and that the generative process is assumed to have no memory of past generated samples

The following crossvalidators can be used in such cases

NOTE

While iid data is a common assumption in machine learning theory it rarely holds in practice If one knows that the samples have been generated using a timedependent process it’s safer to use a timeseries aware crossvalidation scheme Similarly if we know that the generative process has a group structure samples from collected from different subjects experiments measurement devices it safer to use groupwise crossvalidation

Kfold

KFold divides all the samples in k groups of samples called folds if $k=1$ this is equivalent to the Leave One Out strategy of equal sizes if possible The prediction function is learned using $k-1$ folds and the fold left out is used for test

Example of 2fold crossvalidation on a dataset with 4 samples

```
import numpy as np
from sklearn.modelselection import KFold
X = a b c d
kf = KFold(n_splits=2)
for train test in kf.split(X):
    print(s, 'train test')
2 3 0 1
0 1 2 3
```

Here is a visualization of the crossvalidation behavior Note that KFold is not affected by classes or groups

scikitlearn user guide Release 0213

Each fold is constituted by two arrays the first one is related to the training set and the second one to the test set
Thus one can create the trainingtest sets using numpy indexing

```
X nparray0 0 1 1 1 1 2 2
y nparray0 1 0 1
Xtrain Xtest ytrain ytest  Xtrain Xtest ytrain ytest
```

Repeated KFold

RepeatedKFold repeats KFold n times It can be used when one requires to run KFold n times producing
different splits in each repetition

Example of 2fold KFold repeated 2 times

```
import numpy as np
from sklearnmodelselection import RepeatedKFold
X nparray1 2 3 4 1 2 3 4
randomstate 12883823
rkf RepeatedKFoldnplits2 nrepeats2 randomstaterandomstate
for train test inrkfsplitX
prints s train test
```

```
2 3 0 1
0 1 2 3
0 2 1 3
1 3 0 2
```

SimilarlyRepeatedStratifiedKFold repeats Stratified KFold n times with different randomization in each
repetition

Leave One Out LOO

LeaveOneOut or LOO is a simple crossvalidation Each learning set is created by taking all the samples except
one the test set being the sample left out Thus for n samples we have n different training sets and n different tests
set This crossvalidation procedure does not waste much data as only one sample is removed from the training set

```
from sklearnmodelselection import LeaveOneOut
X 1 2 3 4
loo LeaveOneOut
for train test inloosplitX
prints s train test
```

```
1 2 3 0
0 2 3 1
0 1 3 2
0 1 2 3
```

Potential users of LOO for model selection should weigh a few known caveats When compared with nfold cross
validation one builds nmodels from n samples instead of nmodels where n n Moreover each is trained on n-1
samples rather than n-1 In both ways assuming n is not too large and n n LOO is more computationally
expensive than nfold cross validation

In terms of accuracy LOO often results in high variance as an estimator for the test error Intuitively since n-1of
the n samples are used to build each model models constructed from folds are virtually identical to each other and to
the model built from the entire training set

scikitlearn user guide Release 0213

However if the learning curve is steep for the training size in question then 5 or 10 fold cross validation can overestimate the generalization error

As a general rule most authors and empirical evidence suggest that 5 or 10 fold cross validation should be preferred to LOO

References

- <http://www.faq.sorg/faqsaif/neuralnets/part3/section12.html>
- T Hastie R Tibshirani J Friedman The Elements of Statistical Learning Springer 2009
- L Breiman P Spector Submodel selection and evaluation in regression The Xrandom case International Statistical Review 1992
- R Kohavi A Study of CrossValidation and Bootstrap for Accuracy Estimation and Model Selection Intl Jnt Conf AI
- R Bharat Rao G Fung R Rosales On the Dangers of CrossValidation An Experimental Evaluation SIAM 2008
- G James D Witten T Hastie R Tibshirani An Introduction to Statistical Learning Springer 2013

Leave P Out LPO

LeavePOut is very similar to LeaveOneOut as it creates all the possible trainingtest sets by removing $\frac{1}{n}$ samples from the complete set For $\frac{1}{n}$ samples this produces $\frac{1}{n}$

$\frac{1}{n}$ traintest pairs Unlike LeaveOneOut and KFold the test sets will overlap for $\frac{1}{n}$

Example of Leave2Out on a dataset with 4 samples

```
from sklearn.model_selection import LeavePOut
X = np.ones(4)
```

```
lpo = LeavePOut(2)
for train, test in lpo.split(X):
    print(s, train, test)
2 3 0 1
1 3 0 2
1 2 0 3
0 3 1 2
0 2 1 3
0 1 2 3
```

Random permutations crossvalidation aka Shuffle Split

ShuffleSplit

The ShuffleSplit iterator will generate a user defined number of independent train test dataset splits Samples are first shuffled and then split into a pair of train and test sets

It is possible to control the randomness for reproducibility of the results by explicitly seeding the randomstate pseudo random number generator

Here is a usage example

33 Model selection and evaluation 441

```
scikitlearn user guide Release 0213
from sklearn.model_selection import ShuffleSplit
X np.arange(10)
ss ShuffleSplit(n_splits=5, test_size=0.25,
random_state=0)
for train_index, test_index in ss.split(X):
    print(train_index, test_index)
9 1 6 7 3 0 5 2 8 4
2 9 8 0 6 7 4 3 5 1
4 5 1 0 6 9 7 2 3 8
2 7 5 8 0 3 4 6 1 9
4 1 0 6 8 9 3 5 2 7
```

Here is a visualization of the crossvalidation behavior Note that ShuffleSplit is not affected by classes or groups

ShuffleSplit is thus a good alternative to KFold cross validation that allows a finer control on the number of iterations and the proportion of samples on each side of the train test split

Crossvalidation iterators with stratification based on class labels

Some classification problems can exhibit a large imbalance in the distribution of the target classes for instance there could be several times more negative samples than positive samples In such cases it is recommended to use stratified sampling as implemented in StratifiedKFold and StratifiedShuffleSplit to ensure that relative class frequencies is approximately preserved in each train and validation fold

Stratified kfold

StratifiedKFold is a variation of kfold which returns stratified folds each set contains approximately the same percentage of samples of each target class as the complete set

Example of stratified 3fold crossvalidation on a dataset with 10 samples from two slightly unbalanced classes

```
from sklearn.model_selection import StratifiedKFold
X np.ones(10)
y 0 0 0 0 1 1 1 1 1 1
skf StratifiedKFold(n_splits=3)
for train, test, index in skf.split(X, y):
    # ...
442 Chapter 3 User Guide
```


scikitlearn user guide Release 0213

prints s train test

2 3 6 7 8 9 0 1 4 5

0 1 3 4 5 8 9 2 6 7

0 1 2 4 5 6 7 3 8 9

Here is a visualization of the crossvalidation behavior

RepeatedStratifiedKFold can be used to repeat Stratified KFold n times with different randomization in each repetition

Stratified Shuffle Split

StratifiedShuffleSplit is a variation of ShuffleSplit which returns stratified splits iewhich creates splits by preserving the same percentage for each target class as in the complete set

Here is a visualization of the crossvalidation behavior

Crossvalidation iterators for grouped data

The iid assumption is broken if the underlying generative process yield groups of dependent samples

33 Model selection and evaluation 443

scikitlearn user guide Release 0213

Such a grouping of data is domain specific An example would be when there is medical data collected from multiple patients with multiple samples taken from each patient And such data is likely to be dependent on the individual group In our example the patient id for each sample will be its group identifier

In this case we would like to know if a model trained on a particular set of groups generalizes well to the unseen groups To measure this we need to ensure that all the samples in the validation fold come from groups that are not represented at all in the paired training fold

The following crossvalidation splitters can be used to do that The grouping identifier for the samples is specified via thegroups parameter

Group kfold

GroupKFold is a variation of kfold which ensures that the same group is not represented in both testing and training sets For example if the data is obtained from different subjects with several samples persubject and if the model is flexible enough to learn from highly person specific features it could fail to generalize to new subjects GroupKFold makes it possible to detect this kind of overfitting situations

Imagine you have three subjects each with an associated number from 1 to 3

from sklearnmodelselection import GroupKFold

X 01 02 22 24 23 455 58 88 9 10

y a b b b c c c d d d

groups 1 1 1 2 2 2 3 3 3 3

gkf GroupKFoldnplits3

for train test ingkfsplitX y groupsgroups

prints s train test

0 1 2 3 4 5 6 7 8 9

0 1 2 6 7 8 9 3 4 5

3 4 5 6 7 8 9 0 1 2

Each subject is in a different testing fold and the same subject is never in both testing and training Notice that the folds do not have exactly the same size due to the imbalance in the data

Here is a visualization of the crossvalidation behavior

444 Chapter 3 User Guide

Leave One Group Out

LeaveOneGroupOut is a crossvalidation scheme which holds out the samples according to a thirdparty provided array of integer groups This group information can be used to encode arbitrary domain specific predefined cross validation folds

Each training set is thus constituted by all the samples except the ones related to a specific group

For example in the cases of multiple experiments LeaveOneGroupOut can be used to create a crossvalidation based on the different experiments we create a training set using the samples of all the experiments except one

from sklearnmodelselection import LeaveOneGroupOut

X 1 5 10 50 60 70 80

y 0 1 1 2 2 2 2

groups 1 1 2 2 3 3 3

lgo = LeaveOneGroupOut

for train test in lgo.split(X, y, groups):

print('train: %s test: %s' % (train, test))

2 3 4 5 6 0 1

0 1 4 5 6 2 3

0 1 2 3 4 5 6

Another common application is to use time information for instance the groups could be the year of collection of the samples and thus allow for crossvalidation against timebased splits

Leave P Groups Out

LeavePGroupsOut is similar as LeaveOneGroupOut but removes samples related to p groups for each train

ingtest set

Example of Leave2Group Out

from sklearnmodelselection import LeavePGroupsOut

X = np.arange(6)

y = [1, 1, 1, 2, 2, 2]

groups = [1, 1, 2, 2, 3, 3]

lpgo = LeavePGroupsOut(n_groups=2)

for train test in lpgo.split(X, y, groups):

print('train: %s test: %s' % (train, test))

4 5 0 1 2 3

2 3 0 1 4 5

0 1 2 3 4 5

Group Shuffle Split

TheGroupShuffleSplit iterator behaves as a combination of ShuffleSplit and LeavePGroupsOut and generates a sequence of randomized partitions in which a subset of groups are held out for each split

Here is a usage example

from sklearnmodelselection import GroupShuffleSplit

X = [0, 1, 2, 2, 2, 4, 2, 3, 4, 5, 5, 8, 0, 0, 0, 1]

y = ['a', 'b', 'b', 'b', 'c', 'c', 'c', 'a']

```
scikitlearn user guide Release 0213
groups 1 1 2 2 3 3 4 4
gss GroupShuffleSplitn_splits4 testsize05 randomstate0
for train test ingsssplitX y groupsgroups
prints s train test
```

```
0 1 2 3 4 5 6 7
2 3 6 7 0 1 4 5
2 3 4 5 0 1 6 7
4 5 6 7 0 1 2 3
```

Here is a visualization of the crossvalidation behavior

This class is useful when the behavior of LeavePGroupsOut is desired but the number of groups is large enough that generating all possible partitions with $\frac{1}{p}$ groups withheld would be prohibitively expensive In such a scenario GroupShuffleSplit provides a random sample with replacement of the train test splits generated by LeavePGroupsOut

Predefined FoldSplits ValidationSets

For some datasets a predefined split of the data into training and validation fold or into several crossvalidation folds already exists Using PredefinedSplit it is possible to use these folds eg when searching for hyperparameters

For example when using a validation set set the testfold to 0 for all samples that are part of the validation set and to 1 for all other samples

Cross validation of time series data

Time series data is characterised by the correlation between observations that are near in time autocorrelation However classical crossvalidation techniques such as KFold and ShuffleSplit assume the samples are independent and identically distributed and would result in unreasonable correlation between training and testing instances yielding poor estimates of generalisation error on time series data Therefore it is very important to evaluate our model for time series data on the “future” observations least like those that are used to train the model To achieve this one solution is provided by TimeSeriesSplit

446 Chapter 3 User Guide

scikitlearn user guide Release 0213

Time Series Split

TimeSeriesSplit is a variation of kfold which returns first k folds as train set and the $k + 1$ th fold as test set
Note that unlike standard crossvalidation methods successive training sets are supersets of those that come before them Also it adds all surplus data to the first training partition which is always used to train the model
This class can be used to crossvalidate time series data samples that are observed at fixed time intervals

Example of 3split time series crossvalidation on a dataset with 6 samples

```
from sklearn.model_selection import TimeSeriesSplit
```

```
X = np.array([1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4])
```

```
y = np.array([1, 2, 3, 4, 5, 6])
```

```
tscv = TimeSeriesSplit(n_splits=3)
```

```
print(tscv
```

```
TimeSeriesSplit(max_train_size=None, n_splits=3)
```

```
for train, test in tscv.split(X):
```

```
    print('train:', train, 'test:', test)
```

```
0 1 2 3
```

```
0 1 2 3 4
```

```
0 1 2 3 4 5
```

Here is a visualization of the crossvalidation behavior

A note on shuffling

If the data ordering is not arbitrary eg samples with the same class label are contiguous shuffling it first may be essential to get a meaningful cross validation result However the opposite may be true if the samples are not independently and identically distributed For example if samples correspond to news articles and are ordered by their time of publication then shuffling the data will likely lead to a model that is overfit and an inflated validation score it will be tested on samples that are artificially similar close in time to training samples

Some cross validation iterators such as KFold have an inbuilt option to shuffle the data indices before splitting them

Note that

- This consumes less memory than shuffling the data directly
- By default no shuffling occurs including for the stratified K fold cross validation performed by specifying cv=SomeInteger to cross_val_score grid search etc Keep in mind that train_test_split

still returns a random split

scikitlearn user guide Release 0213

- Therandomstate parameter defaults to None meaning that the shuffling will be different every time KFold shuffleTrue is iterated However GridSearchCV will use the same shuffling for each set of parameters validated by a single call to its fit method
- To get identical results for each split set randomstate to an integer

Cross validation and model selection

Cross validation iterators can also be used to directly perform model selection using Grid Search for the optimal hyperparameters of the model This is the topic of the next section Tuning the hyperparameters of an estimator

332 Tuning the hyperparameters of an estimator

Hyperparameters are parameters that are not directly learnt within estimators In scikitlearn they are passed as arguments to the constructor of the estimator classes Typical examples include Ckernel andgamma for Support Vector Classifier alpha for Lasso etc

It is possible and recommended to search the hyperparameter space for the best cross validation score

Any parameter provided when constructing an estimator may be optimized in this manner Specifically to find the names and current values for all parameters for a given estimator use

estimatorgetparams

A search consists of

- an estimator regressor or classifier such as sklearnsvmSVC
- a parameter space
- a method for searching or sampling candidates
- a crossvalidation scheme and
- ascore function

Some models allow for specialized efficient parameter search strategies outlined below Two generic approaches to sampling search candidates are provided in scikitlearn for given values GridSearchCV exhaustively considers all parameter combinations while RandomizedSearchCV can sample a given number of candidates from a parameter space with a specified distribution After describing these tools we detail best practice applicable to both approaches Note that it is common that a small subset of those parameters can have a large impact on the predictive or computation performance of the model while others can be left to their default values It is recommended to read the docstring of the estimator class to get a finer understanding of their expected behavior possibly by reading the enclosed reference to the literature

Exhaustive Grid Search

The grid search provided by GridSearchCV exhaustively generates candidates from a grid of parameter values specified with the paramgrid parameter For instance the following paramgrid

paramgrid

C 1 10 100 1000 kernel linear

C 1 10 100 1000 gamma 0001 00001 kernel rbf

scikitlearn user guide Release 0213

specifies that two grids should be explored one with a linear kernel and C values in 1 10 100 1000 and the second one with an RBF (Radial Basis Function) kernel and the crossproduct of C values ranging in 1 10 100 1000 and gamma values in 0001 00001

TheGridSearchCV instance implements the usual estimator API (Application Programming Interface) when “fitting” it on a dataset all the possible combinations of parameter values are evaluated and the best combination is retained

Examples

- See Parameter estimation using grid search with crossvalidation for an example of Grid Search computation on the digits dataset
- See Sample pipeline for text feature extraction and evaluation for an example of Grid Search coupling parameters from a text documents feature extractor ngram count vectorizer and TFIDF transformer with a classifier here a linear SVM (Support Vector Machine) trained with SGD with either elastic net or L2 penalty using a pipeline Pipeline instance
- See Nested versus nonnested crossvalidation for an example of Grid Search within a cross validation loop on the iris dataset This is the best practice for evaluating the performance of a model with grid search
- See Demonstration of multimetric evaluation on crossvalscore and GridSearchCV for an example of GridSearchCV being used to evaluate multiple metrics simultaneously
- See Balance model complexity and crossvalidated score for an example of using refitcallable in interface inGridSearchCV The example shows how this interface adds certain amount of flexibility in identifying the “best” estimator This interface can also be used in multiple metrics evaluation

Randomized Parameter Optimization

While using a grid of parameter settings is currently the most widely used method for parameter optimization other search methods have more favourable properties RandomizedSearchCV implements a randomized search over parameters where each setting is sampled from a distribution over possible parameter values This has two main benefits over an exhaustive search

- A budget can be chosen independent of the number of parameters and possible values
 - Adding parameters that do not influence the performance does not decrease efficiency
- Specifying how parameters should be sampled is done using a dictionary very similar to specifying parameters for GridSearchCV Additionally a computation budget being the number of sampled candidates or sampling iterations is specified using the niter parameter For each parameter either a distribution over possible values or a list of discrete choices which will be sampled uniformly can be specified

```
C scipystatsexpon scale100 gamma scipystatsexpon scale1
kernel rbf classweightbalanced None
```

This example uses the scipystats module which contains many useful distributions for sampling parameters such as expon gamma uniform orrandint In principle any function can be passed that provides a rvs random variate sample method to sample a value A call to the rvs function should provide independent random samples from possible parameter values on consecutive calls

Warning The distributions in scipystats prior to version scipy 016 do not allow specifying a random state Instead they use the global numpy random state that can be seeded via nprandom seed or set using nprandomsetstate However beginning scikitlearn 018 the sklearn

33 Model selection and evaluation 449

scikitlearn user guide Release 0.21.3

modelselection module sets the random state provided by the user if scipy 0.16 is also available

For continuous parameters such as C above it is important to specify a continuous distribution to take full advantage of the randomization This way increasing niter will always lead to a finer search

Examples

- Comparing randomized search and grid search for hyperparameter estimation compares the usage and efficiency of randomized search and grid search

References

- Bergstra J and Bengio Y Random search for hyperparameter optimization The Journal of Machine Learning Research 2012

Tips for parameter search

Specifying an objective metric

By default parameter search uses the score function of the estimator to evaluate a parameter setting These are sklearn.metrics.accuracy_score for classification and sklearn.metrics.r2_score for regression

For some applications other scoring functions are better suited for example in unbalanced classification the accuracy score is often uninformative An alternative scoring function can be specified via the scoring parameter to GridSearchCV RandomizedSearchCV and many of the specialized crossvalidation tools described below

See The scoring parameter defining model evaluation rules for more details

Specifying multiple metrics for evaluation

GridSearchCV and RandomizedSearchCV allow specifying multiple metrics for the scoring parameter

Multimetric scoring can either be specified as a list of strings of predefined scorer names or a dict mapping the scorer name to the scorer function and/or the predefined scorer names See Using multiple metric evaluation for more details

When specifying multiple metrics the refit parameter must be set to the metric string for which the best params will be found and used to build the best estimator on the whole dataset If the search should not be refit set refit=False Leaving refit to the default value None will result in an error when using multiple metrics

See Demonstration of multimetric evaluation on cross_val_score and GridSearchCV for an example usage

Composite estimators and parameter spaces

Pipeline chaining estimators describes building composite estimators whose parameter space can be searched with these tools

450 Chapter 3 User Guide

scikitlearn user guide Release 0213

Model selection development and evaluation

Model selection by evaluating various parameter settings can be seen as a way to use the labeled data to “train” the parameters of the grid

When evaluating the resulting model it is important to do it on heldout samples that were not seen during the grid search process it is recommended to split the data into a development set to be fed to the GridSearchCV instance and an evaluation set to compute performance metrics

This can be done by using the traintestsplits utility function

Parallelism

GridSearchCV andRandomizedSearchCV evaluate each parameter setting independently Computations can be run in parallel if your OS supports it by using the keyword njobs1 See function signature for more details

Robustness to failure

Some parameter settings may result in a failure to fit one or more folds of the data By default this will cause the entire search to fail even if some parameter settings could be fully evaluated Setting errorscore0 ornp

NaN will make the procedure robust to such failure issuing a warning and setting the score for that fold to 0 or NaN but completing the search

Alternatives to brute force parameter search

Model specific crossvalidation

Some models can fit data for a range of values of some parameter almost as efficiently as fitting the estimator for a single value of the parameter This feature can be leveraged to perform a more efficient crossvalidation used for model selection of this parameter

The most common parameter amenable to this strategy is the parameter encoding the strength of the regularizer In this case we say that we compute the regularization path of the estimator

Here is the list of such models

linearmodelElasticNetCV l1ratio eps Elastic Net model with iterative fitting along a regulariza
tion path

linearmodelLarsCV fitintercept Crossvalidated Least Angle Regression model

linearmodelLassoCV eps nalphas Lasso linear model with iterative fitting along a regulariza
tion path

linearmodelLassoLarsCV fitintercept Crossvalidated Lasso using the LARS algorithm

linearmodelLogisticRegressionCV Cs

Logistic Regression CV (Cross-Validation) aka logit MaxEnt classifier

linearmodelMultiTaskElasticNetCV Multitask L1L2 ElasticNet with builtin crossvalidation

linearmodelMultiTaskLassoCV eps Multitask Lasso model trained with L1L2 mixednorm as
regularizer

linearmodelOrthogonalMatchingPursuitCV Crossvalidated Orthogonal Matching Pursuit model
OMP

linearmodelRidgeCV alphas Ridge regression with builtin crossvalidation

Continued on next page

33 Model selection and evaluation 451

linearmodelRidgeClassifierCV alphas

Ridge classifier with builtin crossvalidation

sklearnlinearmodel ElasticNetCV

classsklearnlinearmodel ElasticNetCV l1ratio05 eps0001 nalphas100 al

phasNone fitinterceptTrue normalizeFalse

precompute'auto' maxiter1000 tol00001

cv'warn' copyXTrue verbose0 njobsNone

positiveFalse randomstateNone selec

tion'cyclic'

Elastic Net model with iterative fitting along a regularization path

See glossary entry for crossvalidation estimator

Read more in the User Guide

Parameters

l1ratio float or array of floats optional float between 0 and 1 passed to ElasticNet scal

ing between l1 and l2 penalties For l1ratio 0 the penalty is an L2 penalty For

l1ratio 1 it is an L1 penalty For 0 l1ratio 1 the penalty is a com

ination of L1 and L2 This parameter can be a list in which case the different values are

tested by crossvalidation and the one giving the best prediction score is used Note that a

good choice of list of values for l1ratio is often to put more values close to 1 ie Lasso

and less close to 0 ie Ridge as in 1 5 7 9 95 99 1

eps float optional Length of the path eps1e3 means that alphas min alphas max

1e3

alphas int optional Number of alphas along the regularization path used for each l1ratio

alphas numpy array optional List of alphas where to compute the models If None alphas are

set automatically

fitintercept boolean whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before

calling fit on an estimator with normalizeFalse

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to

speed up calculations If set to auto let us decide The Gram matrix can also be passed

as argument

maxiter int optional The maximum number of iterations

tol float optional The tolerance for the optimization if the updates are smaller than tol the

optimization code checks the dual gap for optimality and continues until it is smaller than

tol

cv int crossvalidation generator or an iterable optional Determines the crossvalidation split

ting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation

scikitlearn user guide Release 0213

- integer to specify the number of folds
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

copyX boolean optional default True If True X will be copied else it may be overwritten

verbose bool or integer Amount of verbosity

njobs int or None optional defaultNone Number of CPUs to use during the cross valida

tionNone means 1 unless in a joblibparallelbackend context1means using

all processors See Glossary for more details

positive bool optional When set to True forces the coefficients to be positive

randomstate int RandomState instance or None optional default None The seed of the

pseudo random number generator that selects a random feature to update If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom Used when selection 'random'

selection str default 'cyclic' If set to 'random' a random coefficient is updated every iteration

rather than looping over features sequentially by default This setting to 'random' often

leads to significantly faster convergence especially when tol is higher than 1e4

Attributes

alpha float The amount of penalization chosen by cross validation

l1ratio float The compromise between l1 and l2 penalization chosen by cross validation

coef array shape nfeatures ntargets nfeatures Parameter vector w in the cost func

tion formula

intercept float array shape ntargets nfeatures Independent term in the decision func

tion

msepath array shape nl1ratio nalpha nfolds Mean square error for the test set on

each fold varying l1ratio and alpha

alphas numpy array shape nalphas or nl1ratio nalphas The grid of alphas used for

fitting for each l1ratio

niter int number of iterations run by the coordinate descent solver to reach the specified

tolerance for the optimal alpha

See also

enetpath

ElasticNet

Notes

For an example see exampleslinearmodelplotlassomodelselectionpy

33 Model selection and evaluation 453

scikitlearn user guide Release 0213

To avoid unnecessary memory duplication the X argument of the fit method should be directly passed as a Fortran contiguous numpy array

The parameter l1ratio corresponds to alpha in the glmnet R package while alpha corresponds to the lambda parameter in glmnet More specifically the optimization objective is

$$\frac{1}{2} \sum_{i=1}^n \text{samples } y_i - \sum_{j=1}^p \text{Xw}_j^2$$

$$\alpha \text{ l1ratio } w_1$$

$$0.5 \alpha \text{ l1ratio } w_2^2$$

If you are interested in controlling the L1 and L2 penalty separately keep in mind that this is equivalent to

$$\alpha L_1 + \beta L_2$$

for
 α α β and $\text{l1ratio} = \alpha / (\alpha + \beta)$

Examples

```
from sklearn.linear_model import ElasticNetCV
```

```
from sklearn.datasets import make_regression
```

```
X, y = make_regression(n_features=2, random_state=0)
```

```
regr = ElasticNetCV(cv=5, random_state=0)
```

```
regr.fit(X, y)
```

```
ElasticNetCV(alphas=None, copy_X=True, cv=5, eps=0.0001, fit_intercept=True,
```

```
l1_ratio=0.5, max_iter=1000, n_alphas=100, n_jobs=None,
```

```
normalize=False, positive=False, precompute=auto, random_state=0,
```

```
selection='cyclic', tol=0.00001, verbose=0)
```

```
print(regr.alpha_
```

```
0.199
```

```
print(regr.intercept_
```

```
0.398
```

```
print(regr.predict(0, 0)
```

```
0.398
```

Methods

fit(self, X, y) Fit linear model with coordinate descent

get_params(self, deep=True) Get parameters for this estimator

path(X, y, l1_ratio, eps, n_alphas) Compute elastic net path with coordinate descent

predict(self, X) Predict using the linear model

score(self, X, y, sample_weight) Returns the coefficient of determination R² of the prediction

set_params(self, **kwargs) Set the parameters of this estimator

init(self, l1_ratio=0.5, eps=0.0001, n_alphas=100, alphas=None, fit_intercept=True, normal

ize=False, precompute='auto', max_iter=1000, tol=0.00001, cv='warn', copy_X=True, ver

bose=0, n_jobs=None, positive=False, random_state=None, selection='cyclic')

fit(self, X, y)

Fit linear model with coordinate descent

scikitlearn user guide Release 0213

Fit is on grid of alphas and best alpha estimated by crossvalidation

Parameters

Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran contiguous data to avoid unnecessary memory duplication If y is monooutput X can be sparse

yarraylike shape nsamples or nsamples ntargets Target values

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

staticpathXyl1ratio05 eps0001 nalphas100 alphasNone precompute'auto'

XyNone copyXTrue coefinitNone verboseFalse returnniterFalse posi

tiveFalse checkinputTrue params

Compute elastic net path with coordinate descent

The elastic net optimization function varies for mono and multioutputs

For monooutput tasks it is

1 2nsamples y Xw22

alpha l1ratio w1

05alpha1 l1ratio w22

For multioutput tasks it is

1 2 nsamples Y XWFro2

alpha l1ratio W21

05alpha1 l1ratio WFro2

Where

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran

contiguous data to avoid unnecessary memory duplication If yis monooutput then X

can be sparse

yndarray shape nsamples or nsamples noutputs Target values

l1ratio float optional float between 0 and 1 passed to elastic net scaling between l1 and

l2 penalties l1ratio1 corresponds to the Lasso

eps float Length of the path eps1e3 means that alphamin alphamax

1e3

nalphas int optional Number of alphas along the regularization path

33 Model selection and evaluation 455

scikitlearn user guide Release 0213

alphas ndarray optional List of alphas where to compute the models If None alphas are set automatically

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to speed up calculations If set to auto let us decide The Gram matrix can also be passed as argument

Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only when the Gram matrix is precomputed

copyX boolean optional default True If True X will be copied else it may be over written

coefinit array shape nfeatures None The initial values of the coefficients

verbose bool or integer Amount of verbosity

returnniter bool whether to return the number of iterations or not

positive bool default False If set to True forces coefficients to be positive Only allowed whenydim 1

checkinput bool default True Skip input validation checks including the Gram matrix when provided assuming there are handled by the caller when checkinputFalse

params kwargs keyword arguments passed to the coordinate descent solver

Returns

alphas array shape nalphas The alphas along the path where models are computed

coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients along the path

dualgaps array shape nalphas The dual gaps at the end of the optimization for each alpha

niters arraylike shape nalphas The number of iterations taken by the coordinate descent optimizer to reach the specified tolerance for each alpha Is returned when returnniter is set to True

See also

MultiTaskElasticNet

MultiTaskElasticNetCV

ElasticNet

ElasticNetCV

Notes

For an example see `exampleslinearmodelplotlassocoordinatedescentpathpy`

`predictselfX`

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

456 Chapter 3 User Guide

scikitlearn user guide Release 0213

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

sklearnlinearmodel LarsCV

classssklearnlinearmodel LarsCVfitinterceptTrue verboseFalse maxiter500

normalizeTrue precompute'auto'

cv'warn' maxnalphas1000 njobsNone

eps2220446049250313e16 copyXTrue positiveFalse

Crossvalidated Least Angle Regression model

See glossary entry for crossvalidation estimator

Read more in the User Guide

Parameters

fitintercept boolean whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

33 Model selection and evaluation 457

scikitlearn user guide Release 0213

verbose boolean or integer optional Sets the verbosity amount
maxiter integer optional Maximum number of iterations to perform
normalize boolean optional default True This parameter is ignored when fitintercept
is set to False If True the regressors X will be normalized before regression by sub
tracting the mean and dividing by the l2norm If you wish to standardize please use
sklearnpreprocessingStandardScaler before calling fit on an estimator
withnormalizeFalse
precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to
speed up calculations If set to auto let us decide The Gram matrix cannot be passed
as argument since we will use only subsets of X
cvint crossvalidation generator or an iterable optional Determines the crossvalidation split
ting strategy Possible inputs for cv are
• None to use the default 3fold crossvalidation
• integer to specify the number of folds
• CV (Cross-Validation) splitter
• An iterable yielding train test splits as arrays of indices
For integerNone inputs KFold is used
Refer User Guide for the various crossvalidation strategies that can be used here
Changed in version 020 cvdefault value if None will change from 3fold to 5fold in
v022
maxnalphas integer optional The maximum number of points on the path used to compute
the residuals in the crossvalidation
njobs int or None optional defaultNone Number of CPUs to use during the cross valida
tionNone means 1 unless in a joblibparallelbackend context1means using
all processors See Glossary for more details
eps float optional The machineprecision regularization in the computation of the Cholesky
diagonal factors Increase this for very illconditioned systems
copyX boolean optional default True If True X will be copied else it may be overwritten
positive boolean defaultFalse Restrict coefficients to be 0 Be aware that you might
want to remove fitintercept which is set True by default
Deprecated since version 020 The option is broken and deprecated It will be removed in
v022
Attributes
coef array shape nfeatures parameter vector w in the formulation formula
intercept float independent term in decision function
coefpath array shape nfeatures nalphas the varying values of the coefficients along the
path
alpha float the estimated regularization parameter alpha
alphas array shape nalphas the different values of alpha along the path
cvalphas array shape ncvalphas all the values of alpha along the path for the different
folds

scikitlearn user guide Release 0213

msepath array shape nfold ncvalphas the mean square error on leftout for each fold
along the path alpha values given by cvalphas

niter arraylike or int the number of iterations run by Lars with the optimal alpha

See also

larspath LassoLars LassoLarsCV

Examples

```
from sklearn.linear_model import LarsCV
from sklearn.datasets import make_regression
X, y = make_regression(n_samples=200, noise=40, random_state=0)
reg = LarsCV(cv=5)
reg.fit(X, y)
reg.score(X, y)
0.9996
reg.alpha
0.0254
reg.predict(X)
array([1540842])
```

Methods

fitself X, y Fit the model using X, y as training data

get_params self deep Get parameters for this estimator

predict self X Predict using the linear model

score self X, y sample_weight Returns the coefficient of determination R² of the prediction

set_params self params Set the parameters of this estimator

```
init self fit_intercept=True verbose=False max_iter=500 normalize=True precompute='auto' cv='warn' max_n_alphas=1000 n_jobs=None eps=2.220446049250313e-16 copy_X=True positive=False
```

fitself X, y

Fit the model using X, y as training data

Parameters

X arraylike shape n_samples n_features Training data

y arraylike shape n_samples Target values

Returns

self object returns an instance of self

get_params self deep True

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

33 Model selection and evaluation 459

scikitlearn user guide Release 0213

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

460 Chapter 3 User Guide

scikitlearn user guide Release 0213

sklearnlinearmodel LassoCV

classsklearnlinearmodel LassoCVecps0001 nalphas100 alphasNone fitinterceptTrue
normalizeFalse precompute'auto' maxiter1000
tol00001 copyXTrue cv'warn' verboseFalse
njobsNone positiveFalse randomstateNone selec
tion'cyclic'

Lasso linear model with iterative fitting along a regularization path

See glossary entry for crossvalidation estimator

The best model is selected by crossvalidation

The optimization objective for Lasso is

$$\frac{1}{2} \|y - Xw\|_2^2 + \alpha \|w\|_1$$

Read more in the User Guide

Parameters

eps float optional Length of the path eps1e3 means thatalphamin alphamax
1e3

nalphas int optional Number of alphas along the regularization path

alphas numpy array optional List of alphas where to compute the models If None alphas
are set automatically

fitintercept boolean default True whether to calculate the intercept for this model If set to
false no intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when
fitintercept is set to False If True the regressors X will be normalized be
fore regression by subtracting the mean and dividing by the l2norm If you wish to
standardize please use sklearnpreprocessingStandardScaler before
callingfit on an estimator with normalizeFalse

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to
speed up calculations If set to auto let us decide The Gram matrix can also be passed
as argument

maxiter int optional The maximum number of iterations

tolfloat optional The tolerance for the optimization if the updates are smaller than tol the
optimization code checks the dual gap for optimality and continues until it is smaller than
tol

copyX boolean optional default True If True X will be copied else it may be overwritten

cvint crossvalidation generator or an iterable optional Determines the crossvalidation split
ting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation
- integer to specify the number of folds
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

33 Model selection and evaluation 461

scikitlearn user guide Release 0213

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

verbose bool or integer Amount of verbosity

njobs int or None optional defaultNone Number of CPUs to use during the cross validationNone means 1 unless in a joblibparallelbackend context1means using

all processors See Glossary for more details

positive bool optional If positive restrict regression coefficients to be positive

randomstate int RandomState instance or None optional default None The seed of the

pseudo random number generator that selects a random feature to update If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom Used when selection 'random'

selection str default 'cyclic' If set to 'random' a random coefficient is updated every iteration

rather than looping over features sequentially by default This setting to 'random' often

leads to significantly faster convergence especially when tol is higher than 1e4

Attributes

alpha float The amount of penalization chosen by cross validation

coef array shape nfeatures ntargets nfeatures parameter vector w in the cost func

tion formula

intercept float array shape ntargets independent term in decision function

msepath array shape nalphas nfolds mean square error for the test set on each fold

varying alpha

alphas numpy array shape nalphas The grid of alphas used for fitting

dualgap ndarray shape The dual gap at the end of the optimization for the optimal alpha

alpha

niter int number of iterations run by the coordinate descent solver to reach the specified

tolerance for the optimal alpha

See also

larspath

lassopath

LassoLars

Lasso

LassoLarsCV

Notes

For an example see exampleslinearmodelplotlassomodelselectionpy

To avoid unnecessary memory duplication the X argument of the fit method should be directly passed as a

Fortrancontiguous numpy array

462 Chapter 3 User Guide

scikitlearn user guide Release 0213

Examples

```
from sklearn.linear_model import LassoCV
from sklearn.datasets import make_regression
X, y = make_regression(n_samples=100, n_features=10, random_state=0)
reg = LassoCV(cv=5, random_state=0).fit(X, y)
```

reg.score(X, y)

0.9993

reg.predict(X)

array([78.4951])

Methods

fit(X, y) Fit linear model with coordinate descent

get_params(deep=True) Get parameters for this estimator

path(X, y, eps, n_alphas, alphas) Compute Lasso path with coordinate descent

predict(X) Predict using the linear model

score(X, y, sample_weight) Returns the coefficient of determination R² of the prediction

set_params(**kwargs) Set the parameters of this estimator

init(self, eps=0.001, n_alphas=100, alphas=None, fit_intercept=True, normalize=False, precompute='auto', max_iter=1000, tol=0.0001, copy_X=True, cv='warn', verbose=False, n_jobs=None, positive=False, random_state=None, selection='cyclic')

compute('auto', max_iter=1000, tol=0.0001, copy_X=True, cv='warn', verbose=False, n_jobs=None, positive=False, random_state=None, selection='cyclic')

n_jobs=None, positive=False, random_state=None, selection='cyclic')

fit(X, y)

Fit linear model with coordinate descent

Fit is on grid of alphas and best alpha estimated by crossvalidation

Parameters

X: array-like shape (n_samples, n_features) Training data. Pass directly as Fortran

contiguous data to avoid unnecessary memory duplication. If y is monodimensional X can be sparse

y: array-like shape (n_samples) or (n_samples, n_targets) Target values

get_params(deep=True)

Get parameters for this estimator

Parameters

deep: boolean optional. If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params: mapping of string to any. Parameter names mapped to their values

staticpath(X, y, eps=0.001, n_alphas=100, alphas=None, precompute='auto', Xy=None, copy_X=True, coef_init=None, verbose=False, return_niter=False, positive=False)

copy_X=True, coef_init=None, verbose=False, return_niter=False, positive=False)

params

Compute Lasso path with coordinate descent

The Lasso optimization function varies for mono and multioutputs

For monodimensional tasks it is

33 Model selection and evaluation 463

scikitlearn user guide Release 0213

1 2 nsamples y Xw22 alpha w1

For multioutput tasks it is

1 2 nsamples Y XW2Fro alpha W21

Where

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data Pass directly as Fortrancontiguous data to avoid unnecessary memory duplication If yis monooutput thenXcan be sparse

yndarray shape nsamples or nsamples noutputs Target values

eps float optional Length of the path eps1e3 means that alphamin

alphamax 1e3

nalphas int optional Number of alphas along the regularization path

alphas ndarray optional List of alphas where to compute the models If None alphas are set automatically

precompute True False ‘auto’ arraylike Whether to use a precomputed Gram matrix to speed up calculations If set to auto let us decide The Gram matrix can also be passed as argument

Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only when the Gram matrix is precomputed

copyX boolean optional default True If True X will be copied else it may be over written

coefinit array shape nfeatures None The initial values of the coefficients

verbose bool or integer Amount of verbosity

returnniter bool whether to return the number of iterations or not

positive bool default False If set to True forces coefficients to be positive Only allowed whenyndim 1

params kwargs keyword arguments passed to the coordinate descent solver

Returns

alphas array shape nalphas The alphas along the path where models are computed

coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients along the path

dualgaps array shape nalphas The dual gaps at the end of the optimization for each alpha

niters arraylike shape nalphas The number of iterations taken by the coordinate de scent optimizer to reach the specified tolerance for each alpha

464 Chapter 3 User Guide

scikitlearn user guide Release 0213

See also

larspath

Lasso

LassoLars

LassoCV

LassoLarsCV

sklearn.decomposition.sparse_encode

Notes

For an example see `examples/linear_model/plot_lasso_coordinated_descent_path.py`

To avoid unnecessary memory duplication the `X` argument of the fit method should be directly passed as a

Fortran contiguous numpy array

Note that in certain cases the Lars solver may be significantly faster to implement this functionality. In

particular linear interpolation can be used to retrieve model coefficients between the values output by

`larspath`

Examples

Comparing `lassopath` and `larspath` with interpolation

```
X = np.array([2, 31, 23, 54, 43])
```

```
y = np.array([2, 31])
```

Use `lassopath` to compute a coefficient path

```
coef_path = lassopath(X, y, alphas=[5, 1, 5])
```

```
print(coef_path
```

```
0.0 0.046874778
```

```
0.2159048 0.04425765 0.23689075
```

Now use `larspath` and 1D linear interpolation to compute the

same path

```
from sklearn.linear_model import larspath
```

```
alphas, active, coef_path_lars = larspath(X, y, method='lasso')
```

```
from scipy import interpolate
```

```
coef_path_continuous = interpolate.interp1d(alphas, 1,
```

```
coef_path_lars)
```

```
print(coef_path_continuous(5, 1, 5))
```

```
0.0 0.046915237
```

```
0.2159048 0.04425765 0.23668876
```

```
predict_self(X)
```

Predict using the linear model

Parameters

`X`: array-like or sparse matrix shape `(n_samples, n_features)` Samples

Returns

`C`: array shape `(n_samples,)` Returns predicted values

33 Model selection and evaluation 465

scikitlearn user guide Release 0213

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnlinearmodelLassoCV

•Crossvalidation on diabetes Dataset Exercise

•Feature selection using SelectFromModel and LassoCV

•Lasso model selection CrossValidation AIC BIC

sklearnlinearmodel LassoLarsCV

classssklearnlinearmodel LassoLarsCV fitinterceptTrue verboseFalse maxiter500

normalizeTrue precompute'auto'

cv'warn' maxnalphas1000 njobsNone

eps2220446049250313e16 copyXTrue posi

tiveFalse

Crossvalidated Lasso using the LARS algorithm

466 Chapter 3 User Guide

scikitlearn user guide Release 0213

See glossary entry for crossvalidation estimator

The optimization objective for Lasso is

$\frac{1}{2} \sum_{i=1}^n (y_i - \sum_{j=1}^p w_j x_{ij})^2 + \alpha \sum_{j=1}^p |w_j|$

Read more in the User Guide

Parameters

fitintercept boolean whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

verbose boolean or integer optional Sets the verbosity amount

maxiter integer optional Maximum number of iterations to perform

normalize boolean optional default True This parameter is ignored when fitintercept

is set to False If True the regressors X will be normalized before regression by sub

tracting the mean and dividing by the l2norm If you wish to standardize please use

sklearnpreprocessingStandardScaler before calling fit on an estimator

withnormalizeFalse

precompute True False 'auto' Whether to use a precomputed Gram matrix to speed up

calculations If set to auto let us decide The Gram matrix cannot be passed as argument

since we will use only subsets of X

cvint crossvalidation generator or an iterable optional Determines the crossvalidation split

ting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation

- integer to specify the number of folds

- CV (Cross-Validation) splitter

- An iterable yielding train test splits as arrays of indices

For integerNone inputs KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in

v022

maxnalphas integer optional The maximum number of points on the path used to compute

the residuals in the crossvalidation

njobs int or None optional defaultNone Number of CPUs to use during the cross valida

tionNone means 1 unless in a joblibparallelbackend context1means using

all processors See Glossary for more details

eps float optional The machineprecision regularization in the computation of the Cholesky

diagonal factors Increase this for very illconditioned systems

copyX boolean optional default True If True X will be copied else it may be overwritten

positive boolean defaultFalse Restrict coefficients to be ≥ 0 Be aware that you might

want to remove fitintercept which is set True by default Under the positive restriction the

model coefficients do not converge to the ordinaryleastquares solution for small values

of alpha Only coefficients up to the smallest alpha value α_{min} α_{max} α_{1dof}

min when fitpathTrue reached by the stepwise LarsLasso algorithm are typically in

congruence with the solution of the coordinate descent Lasso estimator As a consequence

33 Model selection and evaluation 467

scikitlearn user guide Release 0213

using LassoLarsCV only makes sense for problems where a sparse solution is expected and/or reached

Attributes

coef array shape nfeatures parameter vector w in the formulation formula

intercept float independent term in decision function

coefpath array shape nfeatures nalphas the varying values of the coefficients along the path

alpha float the estimated regularization parameter alpha

alphas array shape nalphas the different values of alpha along the path

cvalphas array shape ncvalphas all the values of alpha along the path for the different folds

msepath array shape nfolds ncvalphas the mean square error on leftout for each fold along the path alpha values given by cvalphas

niter arraylike or int the number of iterations run by Lars with the optimal alpha

See also

larspath LassoLars LarsCV LassoCV

Notes

The object solves the same problem as the LassoCV object However unlike the LassoCV it find the relevant alphas values by itself In general because of this property it will be more stable However it is more fragile to heavily multicollinear datasets

It is more efficient than the LassoCV if only a small number of features are selected compared to the total number for instance if there are very few samples compared to the number of features

Examples

from sklearn.linear_model import LassoLarsCV

from sklearn.datasets import make_regression

X, y = make_regression(n_samples=100, n_features=10, noise=0.1, random_state=0)

reg = LassoLarsCV(cv=5).fit(X, y)

reg.score(X, y)

0.9992

reg.alpha_

0.0484

reg.predict(X)

array([77.8723, ..., -11.9247])

Methods

fit(X, y) Fit the model using X, y as training data

get_params(deep=True) Get parameters for this estimator

predict(X) Predict using the linear model

Continued on next page

468 Chapter 3 User Guide

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selffitinterceptTrue verboseFalse maxiter500 normalizeTrue precom

pute'auto' cv'warn' maxnalphas1000 njobsNone eps2220446049250313e

16copyXTrue positiveFalse

fitselfXy

Fit the model using X y as training data

Parameters

Xarraylike shape nsamples nfeatures Training data

yarraylike shape nsamples Target values

Returns

self object returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

scikitlearn user guide Release 0213

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnlinearmodelLassoLarsCV

•Lasso model selection CrossValidation AIC BIC

sklearnlinearmodel LogisticRegressionCV

classssklearnlinearmodel LogisticRegressionCV Cs10 fitinterceptTrue cv'warn'

dualFalse penalty'l2' scor

ingNone solver'lbfgs' tol00001

maxiter100 classweightNone

njobsNone verbose0 refitTrue in

terceptscaling10 multiclass'warn'

randomstateNone l1ratiosNone

Logistic Regression CV (Cross-Validation) aka logit MaxEnt classifier

See glossary entry for crossvalidation estimator

This class implements logistic regression using liblinear newtoncg sag of lbfgs optimizer The newtoncg sag and lbfgs solvers support only L2 regularization with primal formulation The liblinear solver supports both L1 and L2 regularization with a dual formulation only for the L2 penalty ElasticNet penalty is only supported by the saga solver

For the grid of Csvalues and l1ratios values the best hyperparameter is selected by the crossvalidator

StratifiedKFold but it can be changed using the cvparameter The 'newtoncg' 'sag' 'saga' and 'lbfgs'

solvers can warmstart the coefficients see Glossary

Read more in the User Guide

Parameters

Cslist of floats or int optional default10 Each of the values in Cs describes the inverse of regularization strength If Cs is as an int then a grid of Cs values are chosen in a logarithmic scale between 1e4 and 1e4 Like in support vector machines smaller values specify stronger regularization

fitintercept bool optional defaultTrue Specifies if a constant aka bias or intercept should be added to the decision function

470 Chapter 3 User Guide

scikitlearn user guide Release 0213

cvint or crossvalidation generator optional defaultNone The default crossvalidation generator used is Stratified KFold If an integer is provided then it is the number of folds used See the module sklearnmodelselection module for the list of possible crossvalidation objects

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

dual bool optional defaultFalse Dual or primal formulation Dual formulation is only implemented for l2 penalty with liblinear solver Prefer dualFalse when nsamples nfeatures

penalty str 'l1' 'l2' or 'elasticnet' optional default'l2' Used to specify the norm used in the penalization The 'newtoncg' 'sag' and 'lbfgs' solvers support only l2 penalties 'elasticnet' is only supported by the 'saga' solver

scoring callable or None optional defaultNone A string see model evaluation documentation or a scorer callable object function with signature scorer(estimator, X, y) For a list of scoring functions that can be used look at sklearnmetrics The default scoring option used is 'accuracy'

solver str 'newtoncg' 'lbfgs' 'liblinear' 'sag' 'saga' optional default'lbfgs' Algorithm to use in the optimization problem

- For small datasets 'liblinear' is a good choice whereas 'sag' and 'saga' are faster for large ones
- For multiclass problems only 'newtoncg' 'sag' 'saga' and 'lbfgs' handle multinomial loss 'liblinear' is limited to oneversusrest schemes
- 'newtoncg' 'lbfgs' and 'sag' only handle L2 penalty whereas 'liblinear' and 'saga' handle L1 penalty
- 'liblinear' might be slower in LogisticRegressionCV because it does not handle warm starting

Note that 'sag' and 'saga' fast convergence is only guaranteed on features with approximately the same scale You can preprocess the data with a scaler from sklearnpreprocessing

New in version 017 Stochastic Average Gradient descent solver

New in version 019 SAGA solver

tol float optional default1e4 Tolerance for stopping criteria

max_iter int optional default100 Maximum number of iterations of the optimization algorithm

class_weight dict or 'balanced' optional defaultNone Weights associated with classes in the form {class_label: weight} If not given all classes are supposed to have weight one

The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as $\frac{n_{\text{samples}}}{n_{\text{classes}} \times \text{np.bincount(y)}}$

Note that these weights will be multiplied with sample_weight passed through the fit method if sample_weight is specified

New in version 017 class_weight 'balanced'

33 Model selection and evaluation 471

scikitlearn user guide Release 0213

njobs int or None optional defaultNone Number of CPU cores used during the cross validation loop None means 1 unless in a joblibparallelbackend context1 means using all processors See Glossary for more details

verbose int optional default0 For the 'liblinear' 'sag' and 'lbfgs' solvers set verbose to any positive number for verbosity

refit bool optional defaultTrue If set to True the scores are averaged across all folds and the coefs and the C that corresponds to the best score is taken and a final refit is done using these parameters Otherwise the coefs intercepts and C that correspond to the best scores across folds are averaged

interceptscaling float optional default1 Useful only when the solver 'liblinear' is used and selffitintercept is set to True In this case x becomes x selfinterceptscaling ie a "synthetic" feature with constant value equal to interceptscaling is appended to the instance vector The intercept becomes interceptscaling

syntheticfeatureweight

Note the synthetic feature weight is subject to l1l2 regularization as all other features To lessen the effect of regularization on synthetic feature weight and therefore on the intercept interceptscaling has to be increased

multiclass str 'ovr' 'multinomial' 'auto' optional default'ovr' If the option chosen is 'ovr' then a binary problem is fit for each label For 'multinomial' the loss minimised is the multinomial loss fit across the entire probability distribution even when the data is binary 'multinomial' is unavailable when solver'liblinear' 'auto' selects 'ovr' if the data is binary or if solver'liblinear' and otherwise selects 'multinomial'

New in version 018 Stochastic Average Gradient descent solver for 'multinomial' case Changed in version 020 Default will change from 'ovr' to 'auto' in 022

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

l1ratios list of float or None optional defaultNone The list of ElasticNet mixing parameter with0 l1ratio 1 Only used if penaltyelasticnet A value of 0 is equivalent to using penaltyl2 while 1 is equivalent to using penaltyl1 For0 l1ratio 1 the penalty is a combination of L1 and L2

Attributes

classes array shape nclasses A list of class labels known to the classifier

coef array shape 1 nfeatures or nclasses nfeatures Coefficient of the features in the decision function

coef is of shape 1 nfeatures when the given problem is binary

intercept array shape 1 or nclasses Intercept aka bias added to the decision function

Iffitintercept is set to False the intercept is set to zero intercept is of shape1 when the problem is binary

Cs array shape ncs Array of C ie inverse of regularization parameter values used for crossvalidation

l1ratios array shape nl1ratios Array of l1ratios used for crossvalidation If no l1ratio is used ie penalty is not 'elasticnet' this is set to None

472 Chapter 3 User Guide

scikitlearn user guide Release 0213

coefspaths array shape n folds ncs nfeatures or n folds ncs nfeatures 1 dict with classes as the keys and the path of coefficients obtained during crossvalidating across each fold and then across each Cs after doing an OvR for the corresponding class as values If the 'multiclass' option is set to 'multinomial' then the coefspaths are the coefficients corresponding to each class Each dict value has shape n folds ncs nfeatures or n folds ncs nfeatures 1 depending on whether the intercept is fit or not If penaltyelasticnet the shape is n folds ncs n l1ratios nfeatures or n folds ncs n l1ratios nfeatures 1 scores dict dict with classes as the keys and the values as the grid of scores obtained during crossvalidating each fold after doing an OvR for the corresponding class If the 'multiclass' option given is 'multinomial' then the same scores are repeated across all classes since this is the multinomial class Each dict value has shape n folds ncs or n folds ncs n l1ratios if penaltyelasticnet C array shape nclasses or nclasses 1 Array of C that maps to the best scores across every class If refit is set to False then for each class the best C is the average of the C's that correspond to the best scores for each fold C is of shape nclasses when the problem is binary l1ratio array shape nclasses or nclasses 1 Array of l1ratio that maps to the best scores across every class If refit is set to False then for each class the best l1ratio is the average of the l1ratio's that correspond to the best scores for each fold l1ratio is of shape nclasses when the problem is binary niter array shape nclasses n folds ncs or 1 n folds ncs Actual number of iterations for all classes folds and Cs In the binary or multinomial cases the first dimension is equal to 1 If penaltyelasticnet the shape is nclasses n folds ncs n l1ratios or 1 n folds ncs n l1ratios

See also

LogisticRegression

Examples

```
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegressionCV
X, y = load_iris(return_Xy=True)
clf = LogisticRegressionCV(cv=5, random_state=0)
clf.fit(X, y)
clf.predict(X2)
array([0, 0])
clf.predict_proba(X2).shape
(2, 3)
clf.score(X, y)
0.98
```

Methods

decision_function self X Predict confidence scores for samples

densify self Convert coefficient matrix to dense array format

Continued on next page

33 Model selection and evaluation 473

Table 36 – continued from previous page

`fitself X y sampleweight` Fit the model according to the given training data

`getparams self deep` Get parameters for this estimator

`predict self X` Predict class labels for samples in X

`predictlogproba self X` Log of probability estimates

`predictproba self X` Probability estimates

`score self X y sampleweight` Returns the score using the scoring option on the given test data and labels

`setparams self params` Set the parameters of this estimator

`sparsify self` Convert coefficient matrix to sparse format

`init selfCs10 fitinterceptTrue cv'warn' dualFalse penalty'l2' scoringNone solver'lbfgs' tol00001 maxiter100 classweightNone njobsNone verbose0 refitTrue interceptscaling10 multiclass'warn' randomstateNone l1ratiosNone`

`decisionfunction selfX`

Predict confidence scores for samples

The confidence score for a sample is the signed distance of that sample to the hyperplane

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

array shapensamples if nclasses > 2 else nsamples nclasses Confidence scores per sample class combination In the binary case confidence score for selfclasses1 where 0 means this class would be predicted

`densifyself`

Convert coefficient matrix to dense array format

Converts the coef member back to a numpyndarray This is the default format of coef and is required for fitting so calling this method is only required on models that have previously been sparsified otherwise it is a noop

Returns

self estimator

`fitselfXysampleweightNone`

Fit the model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features yarraylike shape nsamples Target vector relative to X sampleweight arraylike shape nsamples optional Array of weights that are assigned to individual samples If not provided then each sample is given unit weight

Returns

self object

`getparams selfdeepTrue`

Get parameters for this estimator

Parameters

474 Chapter 3 User Guide

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Predicted class label per sample

predictlogproba selfX

Log of probability estimates

The returned estimates for all classes are ordered by the label of classes

Parameters

Xarraylike shape nsamples nfeatures

Returns

Tarraylike shape nsamples nclasses Returns the logprobability of the sample for

each class in the model where classes are ordered as they are in selfclasses

predictproba selfX

Probability estimates

The returned estimates for all classes are ordered by the label of classes

For a multiclass problem if multiclass is set to be “multinomial” the softmax function is used to find the predicted probability of each class Else use a onevsrest approach ie calculate the probability of each class assuming it to be positive using the logistic function and normalize these values across all the classes

Parameters

Xarraylike shape nsamples nfeatures

Returns

Tarraylike shape nsamples nclasses Returns the probability of the sample for each

class in the model where classes are ordered as they are in selfclasses

scoreselfXysampleweightNone

Returns the score using the scoring option on the given test data and labels

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Score of selfpredictX wrt y

33 Model selection and evaluation 475

scikitlearn user guide Release 0.21.3

set_params(self, params)

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines. The latter have parameters of the form `component__parameter` so that it's possible to update each component of a nested object.

Returns

self

sparsify(self)

Convert coefficient matrix to sparse format

Converts the `coef` member to a `scipy.sparse` matrix which for `L1` regularized models can be much more memory and storage efficient than the usual `numpy.ndarray` representation. The `intercept` member is not converted.

Returns

self.estimator

Notes

For non-sparse models (ie when there are not many zeros in `coef`) this may actually increase memory usage so use this method with care. A rule of thumb is that the number of zero elements which can be computed with `coef_0sum` must be more than 50 for this to provide significant benefits.

After calling this method further fitting with the `partial_fit` method if any will not work until you call `densify`.

`sklearn.linear_model.MultiTaskElasticNetCV`

`class sklearn.linear_model.MultiTaskElasticNetCV(l1_ratio=0.5, eps=0.001, n_alphas=100,`

`alpha=None, fit_intercept=True,`

`normalize=False, max_iter=1000,`

`tol=0.0001, cv_warn=True,`

`verbose=0, n_jobs=None, random_state=None,`

`selection='cyclic')`

Multi-task L1/L2 ElasticNet with builtin cross-validation

See glossary entry for cross-validation estimator

The optimization objective for `MultiTaskElasticNet` is

$$\frac{1}{2} \|XW - Y\|_F^2 + \frac{\alpha}{2} \|W\|_1 + \frac{\lambda}{2} \|W\|_2^2$$

where

$$\|W\|_1 = \sum_i \sqrt{\sum_j w_{ij}^2}$$

ie the sum of norm of each row

Read more in the User Guide

Parameters

476 Chapter 3 User Guide

scikitlearn user guide Release 0213

`l1ratio` float or array of floats The ElasticNet mixing parameter with 0 `l1ratio` 1 For `l1ratio` 1 the penalty is an L1L2 penalty For `l1ratio` 0 it is an L2 penalty For 0 `l1ratio` 1 the penalty is a combination of L1L2 and L2 This parameter can be a list in which case the different values are tested by crossvalidation and the one giving the best prediction score is used Note that a good choice of list of values for `l1ratio` is often to put more values close to 1 ie Lasso and less close to 0 ie Ridge as in 1 5 7 9 95 99 1

`eps` float optional Length of the path `eps1e3` means that `alpha_min` `alpha_max` `1e3`

`nalphas` int optional Number of alphas along the regularization path

`alphas` arraylike optional List of alphas where to compute the models If not provided set automatically

`fitintercept` boolean whether to calculate the intercept for this model If set to false no intercept will be used in calculations eg data is expected to be already centered

`normalize` boolean optional default False This parameter is ignored when

`fitintercept` is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the `l2norm` If you wish to

standardize please use `sklearnpreprocessingStandardScaler` before

calling `fit` on an estimator with `normalizeFalse`

`maxiter` int optional The maximum number of iterations

`tol` float optional The tolerance for the optimization if the updates are smaller than `tol` the optimization code checks the dual gap for optimality and continues until it is smaller than `tol`

`cv` int crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for `cv` are

- None to use the default 3fold crossvalidation
- integer to specify the number of folds
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs `KFold` is used

Refer User Guide for the various crossvalidation strategies that can be used here Changed in version 020 `cvdefault` value if None will change from 3fold to 5fold in v022

`copyX` boolean optional default True If True X will be copied else it may be overwritten

`verbose` bool or integer Amount of verbosity

`njobs` int or None optional defaultNone Number of CPUs to use during the cross validation Note that this is used only if multiple values for `l1ratio` are given None means 1 unless in a `joblibparallelbackend` context1means using all processors See Glossary for more details

`randomstate` int `RandomState` instance or None optional default None The seed of the pseudo random number generator that selects a random feature to update If int `randomstate` is the seed used by the random number generator If `RandomState` instance `randomstate` is the random number generator If None the random number generator is the `RandomState` instance used by `nprandom` Used when selection ‘random’

33 Model selection and evaluation 477

scikitlearn user guide Release 0213

selection str default 'cyclic' If set to 'random' a random coefficient is updated every iteration rather than looping over features sequentially by default This setting to 'random' often leads to significantly faster convergence especially when tol is higher than 1e4

Attributes

intercept array shape ntasks Independent term in decision function

coef array shape ntasks nfeatures Parameter vector W in the cost function formula

Note thatcoef stores the transpose of WWT

alpha float The amount of penalization chosen by cross validation

msepath array shape nalphas nfolds or nl1ratio nalphas nfolds mean square error for the test set on each fold varying alpha

alphas numpy array shape nalphas or nl1ratio nalphas The grid of alphas used for fitting for each l1ratio

l1ratio float best l1ratio obtained by crossvalidation

niter int number of iterations run by the coordinate descent solver to reach the specified tolerance for the optimal alpha

See also

MultiTaskElasticNet

ElasticNetCV

MultiTaskLassoCV

Notes

The algorithm used to fit the model is coordinate descent

To avoid unnecessary memory duplication the X argument of the fit method should be directly passed as a

Fortrancontiguous numpy array

Examples

```
from sklearn import linearmodel
clf = linearmodelMultiTaskElasticNetCVcv3
clffit00 1 1 2 2
0 0 1 1 2 2
```

```
MultiTaskElasticNetCValphasNone copyXTrue cv3 eps0001
fitinterceptTrue l1ratio05 maxiter1000 nalphas100
njobsNone normalizeFalse randomstateNone selectioncyclic
tol00001 verbose0
printclcoef
052875032 046958558
052875032 046958558
printclfintercept
000166409 000166409
478 Chapter 3 User Guide
```

Methods

fitself X y Fit linear model with coordinate descent
getparams self deep Get parameters for this estimator
path X y l1ratio eps nalphas Compute elastic net path with coordinate descent
predict self X Predict using the linear model
score self X y sampleweight Returns the coefficient of determination R2 of the prediction
setparams self params Set the parameters of this estimator
init self l1ratio 0.5 eps 0.001 nalphas 100 alphas None fitintercept True normalize False maxiter 1000 tol 0.0001 cv 'warn' copyX True verbose 0 njobs None randomstate None selection 'cyclic' fitself Xy

Fit linear model with coordinate descent
Fit is on grid of alphas and best alpha estimated by crossvalidation

Parameters
Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran contiguous data to avoid unnecessary memory duplication If y is monooutput X can be sparse

yarraylike shape nsamples or nsamples ntargets Target values
getparams self deep True
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns
params mapping of string to any Parameter names mapped to their values
staticpath Xy l1ratio 0.5 eps 0.001 nalphas 100 alphas None precompute 'auto' Xy None copyX True coefinit None verbose False returnniter False positive False checkinput True params
Compute elastic net path with coordinate descent

The elastic net optimization function varies for mono and multioutputs
For monooutput tasks it is
1 2 nsamples y Xw22
alpha l1ratio w1
0.5 alpha 1 l1ratio w22
For multioutput tasks it is
1 2 nsamples Y XWFro2
alpha l1ratio W21
0.5 alpha 1 l1ratio WFro2

Where
33 Model selection and evaluation 479

scikitlearn user guide Release 0213

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran

contiguous data to avoid unnecessary memory duplication If yis monooutput then X

can be sparse

yndarray shape nsamples or nsamples noutputs Target values

l1ratio float optional float between 0 and 1 passed to elastic net scaling between l1 and

l2 penalties l1ratio1 corresponds to the Lasso

eps float Length of the path eps1e3 means that alphamin alphamax

1e3

nalphas int optional Number of alphas along the regularization path

alphas ndarray optional List of alphas where to compute the models If None alphas are set automatically

precompute True False ‘auto’ arraylike Whether to use a precomputed Gram matrix to speed up calculations If set to auto let us decide The Gram matrix can also be passed as argument

Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only when the Gram matrix is precomputed

copyX boolean optional default True If True X will be copied else it may be over written

coefinit array shape nfeatures None The initial values of the coefficients

verbose bool or integer Amount of verbosity

returnniter bool whether to return the number of iterations or not

positive bool default False If set to True forces coefficients to be positive Only allowed

whenyndim 1

checkinput bool default True Skip input validation checks including the Gram matrix when provided assuming there are handled by the caller when checkinputFalse

params kwargs keyword arguments passed to the coordinate descent solver

Returns

alphas array shape nalphas The alphas along the path where models are computed

coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients along the path

dualgaps array shape nalphas The dual gaps at the end of the optimization for each alpha

niters arraylike shape nalphas The number of iterations taken by the coordinate descent optimizer to reach the specified tolerance for each alpha Is returned when returnniter is set to True

See also

480 Chapter 3 User Guide

scikitlearn user guide Release 0.21.3

MultiTaskElasticNet

MultiTaskElasticNetCV

ElasticNet

ElasticNetCV

Notes

For an example see `examples/linear_model/plot_lasso_coordinated_descent_path.py`

`predict_selfX`

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape `nsamples nfeatures` Samples

Returns

Carray shape `nsamples` Returns predicted values

`score_selfXysampleweight`None

Returns the coefficient of determination R^2 of the prediction

The coefficient R^2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R^2 score of 0.0

Parameters

Xarraylike shape `nsamples nfeatures` Test samples For some estimators this may

be a precomputed kernel matrix instead shape `nsamples nsamplesfitted` where

`nsamplesfitted` is the number of samples used in the fitting for the estimator

yarraylike shape `nsamples` or `nsamples noutputs` True values for X

sampleweight arraylike shape `nsamples` optional Sample weights

Returns

score float R^2 of `self.predictX` wrt y

Notes

The R^2 score used when calling `score` on a regressor will use `multioutputuniformaverage`

from version 0.23 to keep consistent with `metricsr2score` This will influence the score

method of all the multioutput regressors except for `MultiOutputRegressor`

To specify the default value manually and avoid the warning please either call `metricsr2score`

directly or make a custom scorer with `metricsmakescorer` the builtin scorer `r2` uses

`multioutputuniformaverage`

`set_params` `self.params`

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form `componentparameter` so that it's possible to update each component

of a nested object

3.3 Model selection and evaluation 481

scikitlearn user guide Release 0213

Returns

self

sklearnlinearmodel MultiTaskLassoCV

classsklearnlinearmodel MultiTaskLassoCV eps0001 nalphas100 alphasNone

fitinterceptTrue normalizeFalse

maxiter1000 tol00001 copyXTrue

cv'warn' verboseFalse njobsNone

randomstateNone selection'cyclic'

Multitask Lasso model trained with L1L2 mixednorm as regularizer

See glossary entry for crossvalidation estimator

The optimization objective for MultiTaskLasso is

$\frac{1}{2} \|Y - XW\|_F^2 + \alpha \|W\|_1$

Where

$\|W\|_1 = \sum_j \sum_i |w_{ij}|$

ie the sum of norm of each row

Read more in the User Guide

Parameters

eps float optional Length of the path eps1e3 means thatalphamin alphamax

1e3

nalphas int optional Number of alphas along the regularization path

alphas arraylike optional List of alphas where to compute the models If not provided set automatically

fitintercept boolean whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before

callingfit on an estimator with normalizeFalse

maxiter int optional The maximum number of iterations

tolfloat optional The tolerance for the optimization if the updates are smaller than tol the optimization code checks the dual gap for optimality and continues until it is smaller than tol

copyX boolean optional default True If True X will be copied else it may be overwritten

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation
- integer to specify the number of folds
- CV (Cross-Validation) splitter

482 Chapter 3 User Guide

scikitlearn user guide Release 0213

- An iterable yielding train test splits as arrays of indices

For integerNone inputs KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

verbose bool or integer Amount of verbosity

njobs int or None optional defaultNone Number of CPUs to use during the cross validation Note that this is used only if multiple values for l1ratio are given None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional default None The seed of the pseudo random number generator that selects a random feature to update If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom Used when selection 'random' selection str default 'cyclic' If set to 'random' a random coefficient is updated every iteration rather than looping over features sequentially by default This setting to 'random' often leads to significantly faster convergence especially when tol is higher than 1e4

Attributes

intercept array shape ntasks Independent term in decision function

coef array shape ntasks nfeatures Parameter vector W in the cost function formula

Note thatcoef stores the transpose of WWT

alpha float The amount of penalization chosen by cross validation

msepath array shape nalphas nfolds mean square error for the test set on each fold varying alpha

alphas numpy array shape nalphas The grid of alphas used for fitting

niter int number of iterations run by the coordinate descent solver to reach the specified tolerance for the optimal alpha

See also

MultiTaskElasticNet

ElasticNetCV

MultiTaskElasticNetCV

Notes

The algorithm used to fit the model is coordinate descent

To avoid unnecessary memory duplication the X argument of the fit method should be directly passed as a Fortrancontiguous numpy array

33 Model selection and evaluation 483

scikitlearn user guide Release 0213

Examples

```
from sklearnlinear import MultiTaskLassoCV
from sklearn.datasets import make_regression
from sklearn.metrics import r2_score
X, y = make_regression(n_targets=2, noise=4, random_state=0)
reg = MultiTaskLassoCV(cv=5, random_state=0)
reg.fit(X, y)
r2_score(y, reg.predict(X))
```

```
reg.alpha_
0.5713
reg.predict(X)
```

Methods

```
fit(X, y) Fit linear model with coordinate descent
get_params(deep=True) Get parameters for this estimator
path(X, y, eps=None, alphas=None) Compute Lasso path with coordinate descent
predict(X) Predict using the linear model
score(X, y, sample_weight=None) Returns the coefficient of determination R2 of the prediction
set_params(**params) Set the parameters of this estimator
init(self, eps=0.001, n_alphas=100, alphas=None, fit_intercept=True, normalize=False, max_iter=1000, tol=0.0001, copy_X=True, cv='warn', verbose=False, n_jobs=None, random_state=None, selection='cyclic')
fit(X, y) Fit linear model with coordinate descent
Fit is on grid of alphas and best alpha estimated by crossvalidation
```

Parameters

```
X: array-like shape (n_samples, n_features) Training data Pass directly as Fortran contiguous data to avoid unnecessary memory duplication If y is mono-output X can be sparse
y: array-like shape (n_samples) or (n_samples, n_targets) Target values
get_params(deep=True) Get parameters for this estimator
```

Parameters

```
deep: boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators
Returns
params: mapping of string to any Parameter names mapped to their values
static_path(X, eps=0.001, n_alphas=100, alphas=None, precompute='auto', Xy=None, copy_X=True, coef_init=None, verbose=False, return_niter=False, positive=False)
params
```

```
Compute Lasso path with coordinate descent
484 Chapter 3 User Guide
```

scikitlearn user guide Release 0213

The Lasso optimization function varies for mono and multioutputs

For monooutput tasks it is

1 2 nsamples y Xw22 alpha w1

For multioutput tasks it is

1 2 nsamples Y XW2Fro alpha W21

Where

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data Pass directly as Fortrancontiguous data to avoid unnecessary memory duplication If yis monooutput thenXcan be sparse

yndarray shape nsamples or nsamples noutputs Target values

eps float optional Length of the path eps1e3 means that alphamin

alphamax 1e3

nalphas int optional Number of alphas along the regularization path

alphas ndarray optional List of alphas where to compute the models If None alphas are set automatically

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to speed up calculations If set to auto let us decide The Gram matrix can also be passed as argument

Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only when the Gram matrix is precomputed

copyX boolean optional default True If True X will be copied else it may be over written

coefinit array shape nfeatures None The initial values of the coefficients

verbose bool or integer Amount of verbosity

returnniter bool whether to return the number of iterations or not

positive bool default False If set to True forces coefficients to be positive Only allowed whenyndim 1

params kwargs keyword arguments passed to the coordinate descent solver

Returns

alphas array shape nalphas The alphas along the path where models are computed

coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients

along the path

dualgaps array shape nalphas The dual gaps at the end of the optimization for each alpha

33 Model selection and evaluation 485

scikitlearn user guide Release 0213

niters arraylike shape nalphas The number of iterations taken by the coordinate de
scent optimizer to reach the specified tolerance for each alpha

See also

larspath

Lasso

LassoLars

LassoCV

LassoLarsCV

sklearn.decomposition.sparse_encode

Notes

For an example see `examples/linear_model/plot_lasso_coordinatedescent_path.py`

To avoid unnecessary memory duplication the `X` argument of the fit method should be directly passed as a
Fortran contiguous numpy array

Note that in certain cases the Lars solver may be significantly faster to implement this functionality In
particular linear interpolation can be used to retrieve model coefficients between the values output by
`larspath`

Examples

Comparing `lassopath` and `larspath` with interpolation

```
X = np.array([2, 31, 23, 54, 43])
y = np.array([2, 31])
# Use lassopath to compute a coefficient path
coef_path = lassopath(X, y, alphas=5)
print(coef_path)
# 0 0 0.46874778
# 0.2159048 0.4425765 0.23689075
```

Now use `larspath` and 1D linear interpolation to compute the
same path

```
from sklearn.linear_model import larspath
alphas = active_coef_path_lars(larspath(X, y, method='lasso'))
from scipy import interpolate
coef_path_continuous = interpolate.interp1d(alphas,
coef_path_lars)
print(coef_path_continuous(5))
# 0 0 0.46915237
# 0.2159048 0.4425765 0.23668876
```

`predict_self(X)`

Predict using the linear model

Parameters

`X` arraylike or sparse matrix shape `n_samples n_features` Samples

Returns

486 Chapter 3 User Guide

scikitlearn user guide Release 0213

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{true\text{mean}}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

sklearnlinearmodel OrthogonalMatchingPursuitCV

classsklearnlinearmodel OrthogonalMatchingPursuitCV copyTrue fitinterceptTrue

normalizeTrue

maxiterNone cv'warn'

njobsNone verboseFalse

Crossvalidated Orthogonal Matching Pursuit model OMP

See glossary entry for crossvalidation estimator

Read more in the User Guide

Parameters

copy bool optional Whether the design matrix X must be copied by the algorithm A false

value is only helpful if X is already Fortranordered otherwise a copy is made anyway

33 Model selection and evaluation 487

scikitlearn user guide Release 0213

fitintercept boolean optional whether to calculate the intercept for this model If set to false no intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default True This parameter is ignored when fitintercept is set to False If True the regressors X will be normalized before regression by subtracting the mean and dividing by the l2norm If you wish to standardize please use sklearnpreprocessingStandardScaler before calling fit on an estimator

withnormalizeFalse

maxiter integer optional Maximum numbers of iterations to perform therefore maximum features to include 10 of nfeatures but at least 5 if available

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation
- integer to specify the number of folds
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

njobs int or None optional defaultNone Number of CPUs to use during the cross validationNone means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

verbose boolean or integer optional Sets the verbosity amount

Attributes

intercept float or array shape ntargets Independent term in decision function

coef array shape nfeatures or ntargets nfeatures Parameter vector w in the problem formulation

nnonzerocoefs int Estimated number of nonzero coefficients giving the best mean squared error over the crossvalidation folds

niter int or arraylike Number of active features across every target for the model refit with the best hyperparameters got by crossvalidating across all folds

See also

orthogonalmp

orthogonalmpgram

larspath

Lars

LassoLars

OrthogonalMatchingPursuit

LarsCV

LassoLarsCV

488 Chapter 3 User Guide

scikitlearn user guide Release 0213

decompositionsparseencode

Examples

```
from sklearn.linear_model import OrthogonalMatchingPursuitCV
from sklearn.datasets import make_regression
X, y = make_regression(n_features=100, n_informative=10,
                      noise=4, random_state=0)
reg = OrthogonalMatchingPursuitCV(cv=5)
reg.fit(X, y)
reg.score(X, y)
0.9991
reg.n_nonzero_coefs
10
reg.predict(X)
```

array(783854)

Methods

fit(self, X, y) Fit the model using X, y as training data

get_params(self, deep=True) Get parameters for this estimator

predict(self, X) Predict using the linear model

score(self, X, y, sample_weight) Returns the coefficient of determination R² of the prediction

set_params(self, **kwargs) Set the parameters of this estimator

init(self, copy=True, fit_intercept=True, normalize=True, max_iter=None, cv='warn', n_jobs=None, verbose=False)

fit(self, X, y)

Fit the model using X, y as training data

Parameters

X: array-like, shape (n_samples, n_features) Training data

y: array-like, shape (n_samples,) Target values. Will be cast to X's dtype if necessary

Returns

self: object returns an instance of self

get_params(self, deep=True)

Get parameters for this estimator

Parameters

deep: boolean, optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params: mapping of string to any Parameter names mapped to their values

predict(self, X)

Predict using the linear model

Parameters

33 Model selection and evaluation 489

scikitlearn user guide Release 0213

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnlinearmodelOrthogonalMatchingPursuitCV

•Orthogonal Matching Pursuit

sklearnlinearmodel RidgeCV

classsklearnlinearmodel RidgeCValphas01 10100 fitinterceptTrue normal

izeFalse scoringNone cvNone gcvmodeNone

storecvvaluesFalse

Ridge regression with builtin crossvalidation

490 Chapter 3 User Guide

scikitlearn user guide Release 0213

See glossary entry for crossvalidation estimator

By default it performs Generalized CrossValidation which is a form of efficient LeaveOneOut cross validation

Read more in the User Guide

Parameters

alphas numpy array of shape nalphas Array of alpha values to try Regularization strength

must be a positive float Regularization improves the conditioning of the problem and re

duces the variance of the estimates Larger values specify stronger regularization Alpha

corresponds to C1 in other linear models such as LogisticRegression or LinearSVC If

using generalized crossvalidation alphas must be positive

fitintercept boolean Whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before

callingfit on an estimator with normalizeFalse

scoring string callable or None optional default None A string see model evaluation doc

umentation or a scorer callable object function with signature scorere estimator

X y If None the negative mean squared error if cv is 'auto' or None ie when using

generalized crossvalidation and r2 score otherwise

cvint crossvalidation generator or an iterable optional Determines the crossvalidation split

ting strategy Possible inputs for cv are

- None to use the efficient LeaveOneOut crossvalidation also known as Generalized

CrossValidation

- integer to specify the number of folds

- CV (Cross-Validation) splitter

- An iterable yielding train test splits as arrays of indices

For integerNone inputs if yis binary or multiclass sklearnmodelselection

StratifiedKfold is used else sklearnmodelselectionKfold is used

Refer User Guide for the various crossvalidation strategies that can be used here

gcvmode None 'auto' 'svd' 'eigen' optional Flag indicating which strategy to use when

performing Generalized CrossValidation Options are

auto use svd ifnsamples nfeatures otherwise use eigen

svd force use of singular value decomposition of X when X is

dense eigenvalue decomposition of XTX when X issparse

eigen force computation via eigendecomposition of XXT

The 'auto' mode is the default and is intended to pick the cheaper option of the two depend

ing on the shape of the training data

storecvvalues boolean defaultFalse Flag indicating if the crossvalidation values corre

sponding to each alpha should be stored in the cvvalues attribute see below This

flag is only compatible with cvNone ie using Generalized CrossValidation

Attributes

33 Model selection and evaluation 491

scikitlearn user guide Release 0213

cvvalues array shape nsamples nalphas or shape nsamples ntargets nalphas optional Crossvalidation values for each alpha if storecvvaluesTrue and cvNone Afterfit has been called this attribute will contain the mean squared errors by default or the values of the losscorefunc function if provided in the constructor

coef array shape nfeatures or ntargets nfeatures Weight vectors

intercept float array shape ntargets Independent term in decision function Set to 00

ifitintercept False

alpha float Estimated regularization parameter

See also

Ridge Ridge regression

RidgeClassifier Ridge classifier

RidgeClassifierCV Ridge classifier with builtin cross validation

Examples

```
from sklearndatasets import loadaddiabetes
from sklearnlinear import RidgeCV
X y loadaddiabetesreturnXy True
clf RidgeCValphas1e3 1e2 1e1 1fitX y
clf.scoreX y
05166
```

Methods

fitself X y sampleweight Fit Ridge regression model

getparams self deep Get parameters for this estimator

predict self X Predict using the linear model

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfalphas01 10100 fitinterceptTrue normalizeFalse scoringNone cvNone

gcvmodeNone storecvvaluesFalse

fitselfXysampleweightNone

Fit Ridge regression model

Parameters

Xarraylike shape nsamples nfeatures Training data If using GCV will be cast to float64 if necessary

yarraylike shape nsamples or nsamples ntargets Target values Will be cast to X's dtype if necessary

sampleweight float or arraylike of shape nsamples Sample weight

Returns

492 Chapter 3 User Guide

self object

Notes

When sampleweight is provided the selected hyperparameter may depend on whether we use generalized crossvalidation cvNone or cv'auto' or another form of crossvalidation because only generalized crossvalidation takes the sample weights into account when computing the validation score

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 0.23 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnlinearmodelRidgeCV

- Face completion with a multioutput estimators
- Effect of transforming the targets in regression model

sklearnlinearmodel RidgeClassifierCV

classsklearnlinearmodel RidgeClassifierCV alphas01 10100 fitinterceptTrue

normalizeFalse scoringNone cvNone

classweightNone storecvvaluesFalse

Ridge classifier with builtin crossvalidation

See glossary entry for crossvalidation estimator

By default it performs Generalized CrossValidation which is a form of efficient LeaveOneOut cross validation Currently only the nfeatures nsamples case is handled efficiently

Read more in the User Guide

Parameters

alphas numpy array of shape nalphas Array of alpha values to try Regularization strength

must be a positive float Regularization improves the conditioning of the problem and reduces the variance of the estimates Larger values specify stronger regularization Alpha corresponds to C1 in other linear models such as LogisticRegression or LinearSVC

fitintercept boolean Whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before

callingfit on an estimator with normalizeFalse

scoring string callable or None optional default None A string see model evaluation doc

umentation or a scorer callable object function with signature scorerestimator

X y

cvint crossvalidation generator or an iterable optional Determines the crossvalidation split

ting strategy Possible inputs for cv are

- None to use the efficient LeaveOneOut crossvalidation
- integer to specify the number of folds
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

494 Chapter 3 User Guide

scikitlearn user guide Release 0213

Refer User Guide for the various crossvalidation strategies that can be used here

classweight dict or 'balanced' optional Weights associated with classes in the form
classlabel weight If not given all classes are supposed to have weight one

The "balanced" mode uses the values of y to automatically adjust weights inversely pro
portional to class frequencies in the input data as nsamples nclasses np
bincounty

storecvvalues boolean defaultFalse Flag indicating if the crossvalidation values corre
sponding to each alpha should be stored in the cvvalues attribute see below This
flag is only compatible with cvNone ie using Generalized CrossValidation

Attributes

cvvalues array shape nsamples ntargets nalphas optional Crossvalidation values
for each alpha if storecvvaluesTrue andcvNone Afterfit has been
called this attribute will contain the mean squared errors by default or the values of the
lossscorefunc function if provided in the constructor

coef array shape 1 nfeatures or ntargets nfeatures Coefficient of the features in the
decision function

coef is of shape 1 nfeatures when the given problem is binary

intercept float array shape ntargets Independent term in decision function Set to 00

ffitintercept False

alpha float Estimated regularization parameter

See also

Ridge Ridge regression

RidgeClassifier Ridge classifier

RidgeCV Ridge regression with builtin cross validation

Notes

For multiclass classification nclass classifiers are trained in a oneversusall approach Concretely this is
implemented by taking advantage of the multivariate response support in Ridge

Examples

```
from sklearndatasets import loadbreastcancer
from sklearnlinear import RidgeClassifierCV
X y loadbreastcancerreturnXy True
clf RidgeClassifierCValphas1e3 1e2 1e1 1fitX y
clf.scoreX y
```

09630

Methods

33 Model selection and evaluation 495

scikitlearn user guide Release 0213

decisionfunction self X Predict confidence scores for samples

fitself X y sampleweight Fit the ridge classifier

getparams self deep Get parameters for this estimator

predict self X Predict class labels for samples in X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init selfalphas01 10100 fitinterceptTrue normalizeFalse scoringNone cvNone

classweightNone storecvvaluesFalse

decisionfunction selfX

Predict confidence scores for samples

The confidence score for a sample is the signed distance of that sample to the hyperplane

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

array shapensamples if nclasses > 2 else nsamples nclasses Confidence scores per sample class combination In the binary case confidence score for selfclasses1 where 0 means this class would be predicted

fitselfXysampleweightNone

Fit the ridge classifier

Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of features When using GCV will be cast to float64 if necessary

yarraylike shape nsamples Target values Will be cast to X's dtype if necessary

sampleweight float or numpy array of shape nsamples Sample weight

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

496 Chapter 3 User Guide

scikitlearn user guide Release 0213

Carray shape nsamples Predicted class label per sample

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Information Criterion

Some models can offer an informationtheoretic closedform formula of the optimal estimate of the regularization

parameter by computing a single regularization path instead of several when using crossvalidation

Here is the list of models benefiting from the Akaike Information Criterion AIC or the Bayesian Information Criterion

BIC for automated model selection

linearmodelLassoLarsIC criterion Lasso model fit with Lars using BIC or AIC for model se

lection

sklearnlinearmodel LassoLarsIC

classsklearnlinearmodel LassoLarsIC criterion'aic' fitinterceptTrue verboseFalse

normalizeTrue precompute'auto' maxiter500

eps2220446049250313e16 copyXTrue posi

tiveFalse

Lasso model fit with Lars using BIC or AIC for model selection

The optimization objective for Lasso is

$\frac{1}{2} \|y - Xw\|_2^2 + \alpha \|w\|_1$

AIC is the Akaike information criterion and BIC is the Bayes Information criterion Such criteria are useful to select the value of the regularization parameter by making a tradeoff between the goodness of fit and the complexity of the model A good model should explain well the data while being simple

Read more in the User Guide

33 Model selection and evaluation 497

Parameters

criterion 'bic' 'aic' The type of criterion to use

fitintercept boolean whether to calculate the intercept for this model If set to false no intercept will be used in calculations eg data is expected to be already centered

verbose boolean or integer optional Sets the verbosity amount

normalize boolean optional default True This parameter is ignored when fitintercept is set to False If True the regressors X will be normalized before regression by subtracting the mean and dividing by the l2norm If you wish to standardize please use sklearnpreprocessingStandardScaler before calling fit on an estimator

withnormalizeFalse

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to speed up calculations If set to auto let us decide The Gram matrix can also be passed as argument

maxiter integer optional Maximum number of iterations to perform Can be used for early stopping

eps float optional The machineprecision regularization in the computation of the Cholesky diagonal factors Increase this for very illconditioned systems Unlike the tol parameter in some iterative optimizationbased algorithms this parameter does not control the tolerance of the optimization

copyX boolean optional default True If True X will be copied else it may be overwritten

positive boolean defaultFalse Restrict coefficients to be 0 Be aware that you might want to remove fitintercept which is set True by default Under the positive restriction the model coefficients do not converge to the ordinaryleastquares solution for small values of alpha Only coefficients up to the smallest alpha value α_{min} are typically in congruence with the solution of the coordinate descent Lasso estimator As a consequence using LassoLarsIC only makes sense for problems where a sparse solution is expected and/or reached

Attributes

coef array shape nfeatures parameter vector w in the formulation formula

intercept float independent term in decision function

alpha float the alpha parameter chosen by the information criterion

niter int number of iterations run by larspath to find the grid of alphas

criterion array shape nalphas The value of the information criteria 'aic' 'bic' across all alphas The alpha which has the smallest information criterion is chosen This value is larger by a factor of nsamples compared to Eqns 215 and 216 in Zou et al 2007

See also

larspath LassoLars LassoLarsCV

Notes

The estimation of the number of degrees of freedom is given by

498 Chapter 3 User Guide

scikitlearn user guide Release 0213

“On the degrees of freedom of the lasso” Hui Zou Trevor Hastie and Robert Tibshirani Ann Statist V olume 35 Number 5 2007 21732192

https://en.wikipedia.org/wiki/Akaike_information_criterion https://en.wikipedia.org/wiki/Bayesian_information_criterion

Examples

```
from sklearn import linear_model
reg = linear_model.LassoLarsIC(criterion=bic)
reg.fit(1 1 0 0 1 1 1 1 1 1 0 1 1 1 1)
```

LassoLarsIC(copy_X=True criterion=bic eps=fit_intercept=True
max_iter=500 normalize=True positive=False precompute=auto
verbose=False
print_reg_coef
0 1 1 1
Methods

fitself X y copy_X Fit the model using X y as training data
get_params self deep Get parameters for this estimator
predict self X Predict using the linear model
score self X y sample_weight Returns the coefficient of determination R2 of the prediction
set_params self params Set the parameters of this estimator
init self criterion='aic' fit_intercept=True verbose=False normalize=True precompute='auto' max_iter=500 eps=2.220446049250313e-16 copy_X=True positive=False
fitself X y copy_X=None
Fit the model using X y as training data

Parameters

X array-like shape (n_samples, n_features) training data
y array-like shape (n_samples) target values Will be cast to X's dtype if necessary
copy_X boolean optional default None If provided this parameter will override the choice of copy_X made at instance creation If True X will be copied else it may be overwritten

Returns

self object returns an instance of self
get_params self deep=True
Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

33 Model selection and evaluation 499

scikitlearn user guide Release 0213

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnlinearmodelLassoLarsIC

•Lasso model selection CrossValidation AIC BIC

500 Chapter 3 User Guide

scikitlearn user guide Release 0213

Out of Bag Estimates

When using ensemble methods base upon bagging ie generating new training sets using sampling with replacement part of the training set remains unused For each classifier in the ensemble a different part of the training set is left out

This left out portion can be used to estimate the generalization error without having to rely on a separate validation set This estimate comes “for free” as no additional data is needed and can be used for model selection

This is currently implemented in the following classes

ensembleRandomForestClassifier A random forest classifier

ensembleRandomForestRegressor A random forest regressor

ensembleExtraTreesClassifier An extratrees classifier

ensembleExtraTreesRegressor nestimators

An extratrees regressor

ensembleGradientBoostingClassifier loss

Gradient Boosting for classification

ensembleGradientBoostingRegressor loss

Gradient Boosting for regression

sklearnensemble RandomForestClassifier

classsklearnensemble RandomForestClassifier nestimators’warn’ crite

rion’gini’ maxdepthNone

minsamplessplit2 minsamplesleaf1

minweightfractionleaf00

maxfeatures’auto’

maxleafnodesNone

minimpuritydecrease00

minimpuritysplitNone bootstrapTrue

oobscoreFalse njobsNone

randomstateNone verbose0

warmstartFalse classweightNone

A random forest classifier

A random forest is a meta estimator that fits a number of decision tree classifiers on various subsamples of the dataset and uses averaging to improve the predictive accuracy and control overfitting The subsample size is always the same as the original input sample size but the samples are drawn with replacement if bootstrapTrue default

Read more in the User Guide

Parameters

nestimators integer optional default10 The number of trees in the forest

Changed in version 020 The default value of nestimators will change from 10 in version 020 to 100 in version 022

criterion string optional default”gini” The function to measure the quality of a split Sup ported criteria are “gini” for the Gini impurity and “entropy” for the information gain Note this parameter is treespecific

maxdepth integer or None optional defaultNone The maximum depth of the tree If None then nodes are expanded until all leaves are pure or until all leaves contain less than 33 Model selection and evaluation 501

scikitlearn user guide Release 0213

minsamplesplit samples

minsamplesplit int float optional default2 The minimum number of samples required to split an internal node

- If int then consider minsamplesplit as the minimum number
- If float then minsamplesplit is a fraction and ceilminsamplesplit

nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions

minsamplesleaf int float optional default1 The minimum number of samples required to be at a leaf node A split point at any depth will only be considered if it leaves at least minsamplesleaf training samples in each of the left and right branches This may have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number
- If float then minsamplesleaf is a fraction and ceilminsamplesleaf

nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the sum total of weights of all the input samples required to be at a leaf node Samples have equal weight when sampleweight is not provided

maxfeatures int float string or None optional default"auto" The number of features to consider when looking for the best split

- If int then consider maxfeatures features at each split
- If float then maxfeatures is a fraction and intmaxfeatures

nfeatures features are considered at each split

- If "auto" then maxfeaturessqrtnfeatures
- If "sqrt" then maxfeaturessqrtnfeatures same as "auto"
- If "log2" then maxfeatureslog2nfeatures
- If None then maxfeaturesnfeatures

Note the search for a split does not stop until at least one valid partition of the node samples

is found even if it requires to effectively inspect more than maxfeatures features

maxleafnodes int or None optional defaultNone Grow trees with maxleafnodes

in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then unlimited number of leaf nodes

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$N_t - N \text{ impurity} - N_t R - N_t \text{ rightimpurity}$$

$$N_t L - N_t \text{ leftimpurity}$$

where N is the total number of samples N_t is the number of samples at the current node

$N_t L$ is the number of samples in the left child and $N_t R$ is the number of samples in

the right child

$N N_t N_t R$ and $N_t L$ all refer to the weighted sum if sampleweight is passed

scikitlearn user guide Release 0213

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of minimpuritydecrease in 019 The default value of minimpuritysplit will change from 1e7 to 0 in 023 and it will be removed in 025 Use minimpuritydecrease instead

bootstrap boolean optional defaultTrue Whether bootstrap samples are used when building trees If False the whole dataset is used to build each tree

oobscore bool defaultFalse Whether to use outofbag samples to estimate the generalization accuracy

njobs int or None optional defaultNone The number of jobs to run in parallel for both fit and predict None means 1 unless in a joblibparallelbackend context

1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

verbose int optional default0 Controls the verbosity when fitting and predicting

warmstart bool optional defaultFalse When set to True reuse the solution of the previous call to fit and add more estimators to the ensemble otherwise just fit a whole new forest See the Glossary

classweight dict list of dicts “balanced” “balancedsubsample” or None optional defaultNone Weights associated with classes in the form classlabel weight

If not given all classes are supposed to have weight one For multioutput problems a list of dicts can be provided in the same order as the columns of y

Note that for multioutput including multilabel weights should be defined for each class of every column in its own dict For example for fourclass multilabel classification weights should be 0 1 1 1 0 1 1 5 0 1 1 1 0 1 1 1 instead of 11 25

31 41

The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as nsamples nclasses np

bincounty

The “balancedsubsample” mode is the same as “balanced” except that weights are computed based on the bootstrap sample for every tree grown

For multioutput the weights of each column of y will be multiplied

Note that these weights will be multiplied with sampleweight passed through the fit method if sampleweight is specified

Attributes

estimators list of DecisionTreeClassifier The collection of fitted subestimators

classes array of shape nclasses or a list of such arrays The classes labels single output problem or a list of arrays of class labels multioutput problem

nclasses int or list The number of classes single output problem or a list containing the number of classes for each output multioutput problem

33 Model selection and evaluation 503

scikitlearn user guide Release 0213

nfeatures int The number of features when fit is performed

noutputs int The number of outputs when fit is performed

featureimportances array of shape nfeatures Return the feature importances
the higher the more important the feature

oobscore float Score of the training dataset obtained using an outofbag estimate

oobdecisionfunction array of shape nsamples nclasses Decision function computed with outofbag estimate on the training set If nestimators is small it might be possible that a data point was never left out during the bootstrap In this case oobdecisionfunction might contain NaN

See also

DecisionTreeClassifier ExtraTreesClassifier

Notes

The default values for the parameters controlling the size of the trees eg maxdepth minsamplesleaf etc lead to fully grown and unpruned trees which can potentially be very large on some data sets To reduce memory consumption the complexity and size of the trees should be controlled by setting those parameter values

The features are always randomly permuted at each split Therefore the best found split may vary even with the same training data maxfeaturesnfeatures andbootstrapFalse if the improvement of the criterion is identical for several splits enumerated during the search of the best split To obtain a deterministic behaviour during fitting randomstate has to be fixed

References

R45f14345c0001

Examples

```
from sklearnensemble import RandomForestClassifier
```

```
from sklearndatasets import makeclassification
```

```
X y = makeclassificationnnsamples1000 nfeatures4
```

```
ninformative2 nredundant0
```

```
randomstate0 shuffle False
```

```
clf = RandomForestClassifiernestimators100 maxdepth2
```

```
randomstate0
```

```
clffitX y
```

```
RandomForestClassifierbootstrapTrue classweightNone criteriongini
```

```
maxdepth2 maxfeaturesauto maxleafnodesNone
```

```
minimpuritydecrease00 minimpuritysplitNone
```

```
minsamplesleaf1 minsamplessplit2
```

```
minweightfractionleaf00 nestimators100 njobsNone
```

```
oobscoreFalse randomstate0 verbose0 warmstartFalse
```

```
printclffeatureimportances
```

```
014205973 076664038 00282433 006305659
```

```
printclfpredict0 0 0 0
```

```
1
```

504 Chapter 3 User Guide

Methods

apply self X Apply trees in the forest to X return leaf indices
decisionpath self X Return the decision path in the forest
fitself X y sampleweight Build a forest of trees from the training set X y
getparams self deep Get parameters for this estimator
predict self X Predict class for X
predictlogproba self X Predict class logprobabilities for X
predictproba self X Predict class probabilities for X
score self X y sampleweight Returns the mean accuracy on the given test data and labels
setparams self params Set the parameters of this estimator
init selfnestimators'warn' criterion'gini' maxdepthNone minsamplesplit2
minsamplesleaf1 minweightfractionleaf00 maxfeatures'auto'
maxleafnodesNone minimpuritydecrease00 minimpuritysplitNone
bootstrapTrue oobscoreFalse njobsNone randomstateNone verbose0
warmstartFalse classweightNone

applyselfX
Apply trees in the forest to X return leaf indices

Parameters
Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csr matrix

Returns
Xleaves arraylike shape nsamples nestimators For each datapoint x in X and for each tree in the forest return the index of the leaf x ends up in
decisionpath selfX
Return the decision path in the forest

New in version 018
Parameters
Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csr matrix

Returns
indicator sparse csr array shape nsamples nnodes Return a node indicator matrix where non zero elements indicates that the samples goes through the nodes
nnodesptr array of size nestimators 1 The columns from indicator
tor nnodesptr in nnodesptr i 1 gives the indicator value for the ith estimator
featureimportances
Return the feature importances the higher the more important the feature
Returns

scikitlearn user guide Release 0213

featureimportances array shape nfeatures The values of this array sum to 1 unless all trees are single node trees consisting of only the root node in which case it will be an array of zeros

fitselfXysampleweightNone

Build a forest of trees from the training set X y

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The training input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse cscmatrix

yarraylike shape nsamples or nsamples noutputs The target values class labels in classification real numbers in regression

sampleweight arraylike shape nsamples or None Sample weights If None then samples are equally weighted Splits that would create child nodes with net zero or negative weight are ignored while searching for a split in each node In the case of classification splits are also ignored if they would result in any single class carrying a negative weight in either child node

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class for X

The predicted class of an input sample is a vote by the trees in the forest weighted by their probability estimates That is the predicted class is the one with highest mean probability estimate across the trees

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csrmatrix

Returns

yarray of shape nsamples or nsamples noutputs The predicted classes

predictlogproba selfX

Predict class logprobabilities for X

The predicted class logprobabilities of an input sample is computed as the log of the mean predicted class probabilities of the trees in the forest

Parameters

506 Chapter 3 User Guide

scikitlearn user guide Release 0213

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csrmatrix

Returns

array of shape nsamples nclasses or a list of noutputs such arrays if noutputs

1 The class probabilities of the input samples The order of the classes corresponds to that in the attribute classes

predict_proba selfX

Predict class probabilities for X

The predicted class probabilities of an input sample are computed as the mean predicted class probabilities of the trees in the forest The class probability of a single tree is the fraction of samples of the same class in a leaf

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csrmatrix

Returns

array of shape nsamples nclasses or a list of noutputs such arrays if noutputs

1 The class probabilities of the input samples The order of the classes corresponds to that in the attribute classes

score selfX sample_weight None

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sample_weight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of self.predictX wrt y

set_params self_params

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form component__parameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearn.ensemble.RandomForestClassifier

•Comparison of Calibration of Classifiers

•Probability Calibration for 3class classification

33 Model selection and evaluation 507

scikitlearn user guide Release 0213

- Classifier comparison
- Inductive Clustering
- Plot class probabilities calculated by the VotingClassifier
- OOB Errors for Random Forests
- Feature transformations with ensembles of trees
- Plot the decision surfaces of ensembles of trees on the iris dataset
- Comparing randomized search and grid search for hyperparameter estimation
- Classification of text documents using sparse features

```
sklearnensemble RandomForestRegressor
classsklearnensemble RandomForestRegressor nestimators'warn' crite
rion'mse' maxdepthNone
minsamplessplit2 minsamplesleaf1
minweightfractionleaf00
maxfeatures'auto' maxleafnodesNone
minimpuritydecrease00
minimpuritysplitNone bootstrapTrue
oobscoreFalse njobsNone
randomstateNone verbose0
warmstartFalse
```

A random forest regressor

A random forest is a meta estimator that fits a number of classifying decision trees on various subsamples of the dataset and uses averaging to improve the predictive accuracy and control overfitting The subsample size is always the same as the original input sample size but the samples are drawn with replacement if bootstrapTrue default

Read more in the User Guide

Parameters

nestimators integer optional default10 The number of trees in the forest
Changed in version 020 The default value of nestimators will change from 10 in version 020 to 100 in version 022

criterion string optional default"mse" The function to measure the quality of a split Supported criteria are "mse" for the mean squared error which is equal to variance reduction as feature selection criterion and "mae" for the mean absolute error

New in version 018 Mean Absolute Error MAE criterion

maxdepth integer or None optional defaultNone The maximum depth of the tree If None then nodes are expanded until all leaves are pure or until all leaves contain less than minsamplessplit samples

minsamplessplit int float optional default2 The minimum number of samples required to split an internal node

- If int then consider minsamplessplit as the minimum number
- If float then minsamplessplit is a fraction and ceilminsamplessplit nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions

508 Chapter 3 User Guide

scikitlearn user guide Release 0213

minsamplesleaf int float optional default1 The minimum number of samples required to be at a leaf node A split point at any depth will only be considered if it leaves at least minsamplesleaf training samples in each of the left and right branches This may have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number
- If float then minsamplesleaf is a fraction and ceilminsamplesleaf nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the sum total of weights of all the input samples required to be at a leaf node Samples have equal weight when sampleweight is not provided

maxfeatures int float string or None optional default"auto" The number of features to consider when looking for the best split

- If int then consider maxfeatures features at each split
- If float then maxfeatures is a fraction and intmaxfeatures nfeatures features are considered at each split

- If "auto" then maxfeaturesnfeatures
- If "sqrt" then maxfeaturesqrtnfeatures
- If "log2" then maxfeatureslog2nfeatures
- If None then maxfeaturesnfeatures

Note the search for a split does not stop until at least one valid partition of the node samples is found even if it requires to effectively inspect more than maxfeatures features

maxleafnodes int or None optional defaultNone Grow trees with maxleafnodes in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then unlimited number of leaf nodes

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$N_t - N \text{ impurity} - \frac{N_t R}{N} - \frac{N_t \text{ right impurity}}{N_t L} - \frac{N_t \text{ left impurity}}{N_t L}$$

where N is the total number of samples Nt is the number of samples at the current node NtL is the number of samples in the left child and NtR is the number of samples in the right child

NNtNtR andNtL all refer to the weighted sum if sampleweight is passed

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of minimpuritydecrease in 019 The default value of minimpuritysplit will change from 1e7 to 0 in 023 and it will be removed in 025 Use minimpuritydecrease instead

scikitlearn user guide Release 0213

bootstrap boolean optional defaultTrue Whether bootstrap samples are used when building trees If False the whole dataset is used to build each tree

oobscore bool optional defaultFalse whether to use outofbag samples to estimate the R2 on unseen data

njobs int or None optional defaultNone The number of jobs to run in parallel for both fit and predict None means 1 unless in a joblibparallelbackend context

1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

verbose int optional default0 Controls the verbosity when fitting and predicting

warmstart bool optional defaultFalse When set to True reuse the solution of the previous call to fit and add more estimators to the ensemble otherwise just fit a whole new forest See the Glossary

Attributes

estimators list of DecisionTreeRegressor The collection of fitted subestimators

featureimportances array of shape nfeatures Return the feature importances the higher the more important the feature

nfeatures int The number of features when fit is performed

noutputs int The number of outputs when fit is performed

oobscore float Score of the training dataset obtained using an outofbag estimate

oobprediction array of shape nsamples Prediction computed with outofbag estimate on the training set

See also

DecisionTreeRegressor ExtraTreesRegressor

Notes

The default values for the parameters controlling the size of the trees eg maxdepth

minsamplesleaf etc lead to fully grown and unpruned trees which can potentially be very large on

some data sets To reduce memory consumption the complexity and size of the trees should be controlled by setting those parameter values

The features are always randomly permuted at each split Therefore the best found split may vary even with

the same training data maxfeaturesnfeatures andbootstrapFalse if the improvement of the

criterion is identical for several splits enumerated during the search of the best split To obtain a deterministic behaviour during fitting randomstate has to be fixed

The default value maxfeaturesauto usesnfeatures rather than nfeatures 3 The latter

was originally suggested in 1 whereas the former was more recently justified empirically in 2

References

Rf91cab2dc4271 Rf91cab2dc4272

510 Chapter 3 User Guide

scikitlearn user guide Release 0213

Examples

```
from sklearnensemble import RandomForestRegressor
from sklearndatasets import makeregression
X y makeregressionnfeatures4 ninformative2
randomstate0 shuffle False
regr RandomForestRegressormaxdepth2 randomstate0
nestimators100
regrfitX y
RandomForestRegressorbootstrapTrue criterionmse maxdepth2
maxfeaturesauto maxleafnodesNone
minimpuritydecrease00 minimpuritysplitNone
minsamplesleaf1 minsamplessplit2
minweightfractionleaf00 nestimators100 njobsNone
oobscoreFalse randomstate0 verbose0 warmstartFalse
printregrfeatureimportances
018146984 081473937 000145312 000233767
printregrpredict0 0 0 0
832987858
```

Methods

```
apply self X Apply trees in the forest to X return leaf indices
decisionpath self X Return the decision path in the forest
fitself X y sampleweight Build a forest of trees from the training set X y
getparams self deep Get parameters for this estimator
predict self X Predict regression target for X
score self X y sampleweight Returns the coefficient of determination R2 of the pre
diction
setparams self params Set the parameters of this estimator
init selfnestimators'warn' criterion'mse' maxdepthNone minsamplessplit2
minsamplesleaf1 minweightfractionleaf00 maxfeatures'auto'
maxleafnodesNone minimpuritydecrease00 minimpuritysplitNone
bootstrapTrue oobscoreFalse njobsNone randomstateNone verbose0
warmstartFalse
applyselfX
Apply trees in the forest to X return leaf indices
Parameters
Xarraylike or sparse matrix shape nsamples nfeatures The input samples Inter
nally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided
it will be converted into a sparse csrmatrix
Returns
Xleaves arraylike shape nsamples nestimators For each datapoint x in X and for
each tree in the forest return the index of the leaf x ends up in
decisionpath selfX
Return the decision path in the forest
33 Model selection and evaluation 511
```

scikitlearn user guide Release 0213

New in version 018

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csr matrix

Returns

indicator sparse csr array shape nsamples nnodes Return a node indicator matrix where non zero elements indicates that the samples goes through the nodes

nnodes ptr array of size nestimators 1 The columns from indicator

tree nnodes ptr i gives the indicator value for the i th estimator

feature importances

Return the feature importances the higher the more important the feature

Returns

feature importances array shape nfeatures The values of this array sum to 1 unless all trees are single node trees consisting of only the root node in which case it will be an array of zeros

fit self X y sample weight None

Build a forest of trees from the training set X y

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The training input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csc matrix

y arraylike shape nsamples or nsamples noutputs The target values class labels in classification real numbers in regression

sample weight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Splits that would create child nodes with net zero or negative weight are ignored while searching for a split in each node In the case of classification splits are also ignored if they would result in any single class carrying a negative

weight in either child node

Returns

self object

get params self deep True

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predict self X

Predict regression target for X

512 Chapter 3 User Guide

scikitlearn user guide Release 0213

The predicted regression target of an input sample is computed as the mean predicted regression targets of the trees in the forest

Parameters

Xarraylike or sparse matrix of shape (nsamples, nfeatures) The input samples Internally its dtype will be converted to dtype np.float32 If a sparse matrix is provided it will be converted into a sparse csr matrix

Returns

array of shape (nsamples, noutputs) The predicted values

score self.X sample_weight None

Returns the coefficient of determination R² of the prediction

The coefficient R² is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R² score of 0.0

Parameters

Xarraylike shape (nsamples, nfeatures) Test samples For some estimators this may

be a precomputed kernel matrix instead shape (nsamples, nsamplesfitted) where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape (nsamples, noutputs) True values for X

sample_weight arraylike shape (nsamples, optional) Sample weights

Returns

score float R² of self.predict(X) wrt y

Notes

The R² score used when calling score on a regressor will use multioutputuniformaverage

from version 0.23 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams self.params

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form component.parameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnensembleRandomForestRegressor

- Prediction Latency
- Plot individual and voting regression predictions

33 Model selection and evaluation 513

scikitlearn user guide Release 0213

- Comparing random forests and the multioutput meta estimator
- Imputing missing values before building an estimator

sklearnensemble ExtraTreesClassifier

classsklearnensemble ExtraTreesClassifier nestimators'warn' crite

rion'gini' maxdepthNone

minsamplesplit2 minsamplesleaf1

minweightfractionleaf00

maxfeatures'auto' maxleafnodesNone

minimpuritydecrease00

minimpuritysplitNone bootstrapFalse

oobscoreFalse njobsNone

randomstateNone verbose0

warmstartFalse classweightNone

An extratrees classifier

This class implements a meta estimator that fits a number of randomized decision trees aka extratrees on various subsamples of the dataset and uses averaging to improve the predictive accuracy and control overfitting

Read more in the User Guide

Parameters

nestimators integer optional default10 The number of trees in the forest

Changed in version 020 The default value of nestimators will change from 10 in

version 020 to 100 in version 022

criterion string optional default"gini" The function to measure the quality of a split Sup

ported criteria are "gini" for the Gini impurity and "entropy" for the information gain

maxdepth integer or None optional defaultNone The maximum depth of the tree If

None then nodes are expanded until all leaves are pure or until all leaves contain less than

minsamplessplit samples

minsamplessplit int float optional default2 The minimum number of samples required

to split an internal node

- If int then consider minsamplessplit as the minimum number
- If float then minsamplessplit is a fraction and ceilminsamplessplit

nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions

minsamplesleaf int float optional default1 The minimum number of samples required

to be at a leaf node A split point at any depth will only be considered if it leaves at least

minsamplesleaf training samples in each of the left and right branches This may

have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number
- If float then minsamplesleaf is a fraction and ceilminsamplesleaf

nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the

sum total of weights of all the input samples required to be at a leaf node Samples have

equal weight when sampleweight is not provided

514 Chapter 3 User Guide

scikitlearn user guide Release 0213

maxfeatures int float string or None optional default"auto" The number of features to consider when looking for the best split

- If int then consider maxfeatures features at each split
- If float then maxfeatures is a fraction and intmaxfeatures nfeatures features are considered at each split
- If "auto" then maxfeatures \sqrt{n} features
- If "sqrt" then maxfeatures \sqrt{n} features
- If "log2" then maxfeatures $\log_2 n$ features
- If None then maxfeaturesnfeatures

Note the search for a split does not stop until at least one valid partition of the node samples

is found even if it requires to effectively inspect more than maxfeatures features

maxleafnodes int or None optional defaultNone Grow trees with maxleafnodes

in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then

unlimited number of leaf nodes

minimpuritydecrease float optional default0 A node will be split if this split induces

a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$Nt \cdot N \cdot \text{impurity} - NtR \cdot Nt \cdot \text{rightimpurity}$$

$$- NtL \cdot Nt \cdot \text{leftimpurity}$$

whereNis the total number of samples Nt is the number of samples at the current node

NtL is the number of samples in the left child and NtR is the number of samples in

the right child

NNtNtR andNtL all refer to the weighted sum if sampleweight is passed

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node

will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of

minimpuritydecrease in 019 The default value of minimpuritysplit

will change from 1e7 to 0 in 023 and it will be removed in 025 Use

minimpuritydecrease instead

bootstrap boolean optional defaultFalse Whether bootstrap samples are used when build

ing trees If False the whole dataset is used to build each tree

oobscore bool optional defaultFalse Whether to use outofbag samples to estimate the

generalization accuracy

njobs int or None optional defaultNone The number of jobs to run in parallel for both

fit andpredict None means 1 unless in a joblibparallelbackend context

1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

verbose int optional default0 Controls the verbosity when fitting and predicting

33 Model selection and evaluation 515

scikitlearn user guide Release 0213

warmstart bool optional defaultFalse When set to True reuse the solution of the previous call to fit and add more estimators to the ensemble otherwise just fit a whole new forest See the Glossary

classweight dict list of dicts “balanced” “balancedsubsample” or None optional defaultNone Weights associated with classes in the form classlabel weight

If not given all classes are supposed to have weight one For multioutput problems a list of dicts can be provided in the same order as the columns of y

Note that for multioutput including multilabel weights should be defined for each class of every column in its own dict For example for fourclass multilabel classification weights should be 0 1 1 1 0 1 1 5 0 1 1 1 0 1 1 1 instead of 11 25

31 41

The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as nsamples nclasses np

bincounty

The “balancedsubsample” mode is the same as “balanced” except that weights are computed based on the bootstrap sample for every tree grown

For multioutput the weights of each column of y will be multiplied

Note that these weights will be multiplied with sampleweight passed through the fit method if sampleweight is specified

Attributes

estimators list of DecisionTreeClassifier The collection of fitted subestimators

classes array of shape nclasses or a list of such arrays The classes labels single output problem or a list of arrays of class labels multioutput problem

nclasses int or list The number of classes single output problem or a list containing the number of classes for each output multioutput problem

featureimportances array of shape nfeatures Return the feature importances the higher the more important the feature

nfeatures int The number of features when fit is performed

noutputs int The number of outputs when fit is performed

oobscore float Score of the training dataset obtained using an outofbag estimate

oobdecisionfunction array of shape nsamples nclasses Decision function computed with outofbag estimate on the training set If nestimators is small it might be possible that a data point was never left out during the bootstrap In this case

oobdecisionfunction might contain NaN

See also

sklearn.tree.ExtraTreeClassifier Base classifier for this ensemble

RandomForestClassifier Ensemble Classifier based on trees with optimal splits

Notes

The default values for the parameters controlling the size of the trees eg maxdepth

minsamplesleaf etc lead to fully grown and unpruned trees which can potentially be very large on

516 Chapter 3 User Guide

scikitlearn user guide Release 0213

some data sets To reduce memory consumption the complexity and size of the trees should be controlled by setting those parameter values

References

Rc8f28bfad63f1

Methods

apply self X Apply trees in the forest to X return leaf indices

decisionpath self X Return the decision path in the forest

fitself X y sampleweight Build a forest of trees from the training set X y

getparams self deep Get parameters for this estimator

predict self X Predict class for X

predictlogproba self X Predict class logprobabilities for X

predictproba self X Predict class probabilities for X

score self X y sampleweight Returns the mean accuracy on the given test data and

labels

setparams self params Set the parameters of this estimator

init selfnestimators'warn' criterion'gini' maxdepthNone minsamplessplit2

minsamplesleaf1 minweightfractionleaf00 maxfeatures'auto'

maxleafnodesNone minimpuritydecrease00 minimpuritysplitNone boot

strapFalse oobscoreFalse njobsNone randomstateNone verbose0

warmstartFalse classweightNone

applyselfX

Apply trees in the forest to X return leaf indices

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype=np.float32 If a sparse matrix is provided it will be converted into a sparse csrmatrix

Returns

Xleaves arraylike shape nsamples nestimators For each datapoint x in X and for each tree in the forest return the index of the leaf x ends up in

decisionpath selfX

Return the decision path in the forest

New in version 018

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype=np.float32 If a sparse matrix is provided it will be converted into a sparse csrmatrix

Returns

indicator sparse csr array shape nsamples nnodes Return a node indicator matrix where non zero elements indicates that the samples goes through the nodes

scikitlearn user guide Release 0213

nnodesptr array of size nestimators 1 The columns from indica
tornnodesptrinnodesptri1 gives the indicator value for the ith estimator
featureimportances

Return the feature importances the higher the more important the feature
Returns

featureimportances array shape nfeatures The values of this array sum to 1 unless
all trees are single node trees consisting of only the root node in which case it will be an
array of zeros

fitselfXysampleweightNone

Build a forest of trees from the training set X y

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The training input sam
ples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix
is provided it will be converted into a sparse cscmatrix

yarraylike shape nsamples or nsamples noutputs The target values class labels
in classification real numbers in regression

sampleweight arraylike shape nsamples or None Sample weights If None then
samples are equally weighted Splits that would create child nodes with net zero or neg
ative weight are ignored while searching for a split in each node In the case of classifi
cation splits are also ignored if they would result in any single class carrying a negative
weight in either child node

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class for X

The predicted class of an input sample is a vote by the trees in the forest weighted by their probability
estimates That is the predicted class is the one with highest mean probability estimate across the trees

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Inter
nally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided
it will be converted into a sparse csrmatrix

Returns

yarray of shape nsamples or nsamples noutputs The predicted classes

518 Chapter 3 User Guide

scikitlearn user guide Release 0213

predictlogproba selfX

Predict class logprobabilities for X

The predicted class logprobabilities of an input sample is computed as the log of the mean predicted class probabilities of the trees in the forest

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csr matrix

Returns

array of shape nsamples nclasses or a list of noutputs such arrays if noutputs

1 The class probabilities of the input samples The order of the classes corresponds to that in the attribute classes

predictproba selfX

Predict class probabilities for X

The predicted class probabilities of an input sample are computed as the mean predicted class probabilities of the trees in the forest The class probability of a single tree is the fraction of samples of the same class in a leaf

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csr matrix

Returns

array of shape nsamples nclasses or a list of noutputs such arrays if noutputs

1 The class probabilities of the input samples The order of the classes corresponds to that in the attribute classes

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

33 Model selection and evaluation 519

scikitlearn user guide Release 0213

Examples using sklearnensembleExtraTreesClassifier

- Pixel importances with a parallel forest of trees
- Feature importances with forests of trees
- Hashing feature transformation using Totally Random Trees
- Plot the decision surfaces of ensembles of trees on the iris dataset

sklearnensemble ExtraTreesRegressor

classsklearnensemble ExtraTreesRegressor nestimators'warn' crite

rion'mse' maxdepthNone

minsamplessplit2 minsamplesleaf1

minweightfractionleaf00

maxfeatures'auto' maxleafnodesNone

minimpuritydecrease00

minimpuritysplitNone bootstrapFalse

oobscoreFalse njobsNone

randomstateNone verbose0

warmstartFalse

An extratrees regressor

This class implements a meta estimator that fits a number of randomized decision trees aka extratrees on

various subsamples of the dataset and uses averaging to improve the predictive accuracy and control overfitting

Read more in the User Guide

Parameters

nestimators integer optional default10 The number of trees in the forest

Changed in version 020 The default value of nestimators will change from 10 in

version 020 to 100 in version 022

criterion string optional default"mse" The function to measure the quality of a split Sup
ported criteria are "mse" for the mean squared error which is equal to variance reduction as
feature selection criterion and "mae" for the mean absolute error

New in version 018 Mean Absolute Error MAE criterion

maxdepth integer or None optional defaultNone The maximum depth of the tree If
None then nodes are expanded until all leaves are pure or until all leaves contain less than

minsamplessplit samples

minsamplessplit int float optional default2 The minimum number of samples required
to split an internal node

- If int then consider minsamplessplit as the minimum number
- If float then minsamplessplit is a fraction and ceilminsamplessplit

nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions

minsamplesleaf int float optional default1 The minimum number of samples required
to be at a leaf node A split point at any depth will only be considered if it leaves at least

minsamplesleaf training samples in each of the left and right branches This may
have the effect of smoothing the model especially in regression

520 Chapter 3 User Guide

scikitlearn user guide Release 0213

- If int then consider minsamplesleaf as the minimum number
- If float then minsamplesleaf is a fraction and ceilminsamplesleaf

nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the sum total of weights of all the input samples required to be at a leaf node Samples have equal weight when sampleweight is not provided

maxfeatures int float string or None optional default"auto" The number of features to consider when looking for the best split

- If int then consider maxfeatures features at each split
- If float then maxfeatures is a fraction and intmaxfeatures
- nfeatures features are considered at each split
- If "auto" then maxfeaturesnfeatures
- If "sqrt" then maxfeaturessqrtnfeatures
- If "log2" then maxfeatureslog2nfeatures
- If None then maxfeaturesnfeatures

Note the search for a split does not stop until at least one valid partition of the node samples is found even if it requires to effectively inspect more than maxfeatures features

maxleafnodes int or None optional defaultNone Grow trees with maxleafnodes

in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then unlimited number of leaf nodes

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$N_t \cdot N \cdot \text{impurity} - N_t R \cdot N_t \text{rightimpurity}$$

$$- N_t L \cdot N_t \text{leftimpurity}$$

where N is the total number of samples Nt is the number of samples at the current node

NtL is the number of samples in the left child and NtR is the number of samples in

the right child

NNtNtR andNtL all refer to the weighted sum if sampleweight is passed

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node

will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of

minimpuritydecrease in 019 The default value of minimpuritysplit

will change from 1e7 to 0 in 023 and it will be removed in 025 Use

minimpuritydecrease instead

bootstrap boolean optional defaultFalse Whether bootstrap samples are used when build

ing trees If False the whole dataset is used to build each tree

oobscore bool optional defaultFalse Whether to use outofbag samples to estimate the

R2 on unseen data

33 Model selection and evaluation 521

scikitlearn user guide Release 0213

njobs int or None optional defaultNone The number of jobs to run in parallel for both fit and predict None means 1 unless in a joblibparallelbackend context

1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

verbose int optional default0 Controls the verbosity when fitting and predicting

warmstart bool optional defaultFalse When set to True reuse the solution of the pre

vious call to fit and add more estimators to the ensemble otherwise just fit a whole new

forest See the Glossary

Attributes

estimators list of DecisionTreeRegressor The collection of fitted subestimators

featureimportances array of shape nfeatures Return the feature importances

the higher the more important the feature

nfeatures int The number of features

noutputs int The number of outputs

oobscore float Score of the training dataset obtained using an outofbag estimate

oobprediction array of shape nsamples Prediction computed with outofbag estimate on the training set

See also

sklearnmtreeExtraTreeRegressor Base estimator for this ensemble

RandomForestRegressor Ensemble regressor using trees with optimal splits

Notes

The default values for the parameters controlling the size of the trees eg maxdepth

minsamplesleaf etc lead to fully grown and unpruned trees which can potentially be very large on

some data sets To reduce memory consumption the complexity and size of the trees should be controlled by setting those parameter values

References

Ra7d0c8995fbc1

Methods

apply self X Apply trees in the forest to X return leaf indices

decisionpath self X Return the decision path in the forest

fitself X y sampleweight Build a forest of trees from the training set X y

getparams self deep Get parameters for this estimator

predict self X Predict regression target for X

Continued on next page

522 Chapter 3 User Guide

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator
init selfnestimators'warn' criterion'mse' maxdepthNone minsamplessplit2
minsamplesleaf1 minweightfractionleaf00 maxfeatures'auto'
maxleafnodesNone minimpuritydecrease00 minimpuritysplitNone boot
strapFalse oobscoreFalse njobsNone randomstateNone verbose0
warmstartFalse
applyselfX

Apply trees in the forest to X return leaf indices

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csrmatrix

Returns

Xleaves arraylike shape nsamples nestimators For each datapoint x in X and for each tree in the forest return the index of the leaf x ends up in

decisionpath selfX

Return the decision path in the forest

New in version 018

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse csrmatrix

Returns

indicator sparse csr array shape nsamples nnodes Return a node indicator matrix where non zero elements indicates that the samples goes through the nodes

nnodesptr array of size nestimators 1 The columns from indicator

tor nnodesptr in nnodesptr i gives the indicator value for the ith estimator

featureimportances

Return the feature importances the higher the more important the feature

Returns

featureimportances array shape nfeatures The values of this array sum to 1 unless all trees are single node trees consisting of only the root node in which case it will be an array of zeros

fitselfXysampleweightNone

Build a forest of trees from the training set X y

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The training input samples Internally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided it will be converted into a sparse cscmatrix

scikitlearn user guide Release 0213

yarraylike shape nsamples or nsamples noutputs The target values class labels in classification real numbers in regression

sampleweight arraylike shape nsamples or None Sample weights If None then samples are equally weighted Splits that would create child nodes with net zero or negative weight are ignored while searching for a split in each node In the case of classification splits are also ignored if they would result in any single class carrying a negative weight in either child node

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict regression target for X

The predicted regression target of an input sample is computed as the mean predicted regression targets of the trees in the forest

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally its dtype will be converted to dtype=np.float32 If a sparse matrix is provided it will be converted into a sparse csrmatrix

Returns

yarray of shape nsamples or nsamples noutputs The predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

524 Chapter 3 User Guide

scikitlearn user guide Release 0213

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage setparams selfparams

Set the parameters of this estimator The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self Examples using sklearnensembleExtraTreesRegressor

- Face completion with a multioutput estimators
- Imputing missing values with variants of IterativeImputer

sklearnensemble GradientBoostingClassifier classsklearnensemble GradientBoostingClassifier loss'deviance' learningrate01 nestimators100 subsample10 criterion'friedmanmse' minsamplesplit2 minsamplesleaf1 minweightfractionleaf00 maxdepth3 minimpuritydecrease00 minimpuritysplitNone initNone randomstateNone maxfeaturesNone verbose0 maxleafnodesNone warmstartFalse pre sort'auto' validationfraction01 niternochangeNone tol00001

Gradient Boosting for classification

GB builds an additive model in a forward stagewise fashion it allows for the optimization of arbitrary differentiable loss functions In each stage nclasses regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function Binary classification is a special case where only a single regression tree is induced

Read more in the User Guide

Parameters

33 Model selection and evaluation 525

scikitlearn user guide Release 0213

loss ‘deviance’ ‘exponential’ optional default ‘deviance’ loss function to be optimized
‘deviance’ refers to deviance logistic regression for classification with probabilistic out
puts For loss ‘exponential’ gradient boosting recovers the AdaBoost algorithm
learningrate float optional default 0.1 learning rate shrinks the contribution of each tree
by learningrate There is a tradeoff between learningrate and nestimators
nestimators int default 100 The number of boosting stages to perform Gradient boosting
is fairly robust to overfitting so a large number usually results in better performance
subsample float optional default 0.1 The fraction of samples to be used for fitting the in
dividual base learners If smaller than 10 this results in Stochastic Gradient Boosting
subsample interacts with the parameter nestimators Choosing subsample
0.1 leads to a reduction of variance and an increase in bias
criterion string optional default “friedmanmse” The function to measure the quality of
a split Supported criteria are “friedmanmse” for the mean squared error with improve
ment score by Friedman “mse” for mean squared error and “mae” for the mean absolute
error The default value of “friedmanmse” is generally the best as it can provide a better
approximation in some cases

New in version 0.18

min_samples_split int float optional default 2 The minimum number of samples required
to split an internal node

- If int then consider min_samples_split as the minimum number
- If float then min_samples_split is a fraction and ceil(min_samples_split
n_samples) are the minimum number of samples for each split

Changed in version 0.18 Added float values for fractions

min_samples_leaf int float optional default 1 The minimum number of samples required
to be at a leaf node A split point at any depth will only be considered if it leaves at least
min_samples_leaf training samples in each of the left and right branches This may
have the effect of smoothing the model especially in regression

- If int then consider min_samples_leaf as the minimum number
- If float then min_samples_leaf is a fraction and ceil(min_samples_leaf
n_samples) are the minimum number of samples for each node

Changed in version 0.18 Added float values for fractions

min_weight_fraction_leaf float optional default 0 The minimum weighted fraction of the
sum total of weights of all the input samples required to be at a leaf node Samples have
equal weight when sample_weight is not provided

max_depth integer optional default 3 maximum depth of the individual regression estima
tors The maximum depth limits the number of nodes in the tree Tune this parameter for
best performance the best value depends on the interaction of the input variables

min_impurity_decrease float optional default 0 A node will be split if this split induces
a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$N_t \cdot N \cdot \text{impurity} - N_t R \cdot N_t \cdot \text{rightimpurity}$$

$$N_t L \cdot N_t \cdot \text{leftimpurity}$$

526 Chapter 3 User Guide

scikitlearn user guide Release 0213

where N is the total number of samples N_t is the number of samples at the current node
 N_{tL} is the number of samples in the left child and N_{tR} is the number of samples in the right child
 NN_tN_{tR} and N_{tL} all refer to the weighted sum if sampleweight is passed
New in version 019
minimpuritysplit float default $1e7$ Threshold for early stopping in tree growth A node will split if its impurity is above the threshold otherwise it is a leaf
Deprecated since version 019 minimpuritysplit has been deprecated in favor of minimpuritydecrease in 019 The default value of minimpuritysplit will change from $1e7$ to 0 in 023 and it will be removed in 025 Use minimpuritydecrease instead
init estimator or 'zero' optional default None An estimator object that is used to compute the initial predictions init has to provide fit and predict_proba If 'zero' the initial raw predictions are set to zero By default a DummyEstimator predicting the classes priors is used
randomstate int RandomState instance or None optional default None If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random
maxfeatures int float string or None optional default None The number of features to consider when looking for the best split
• If int then consider maxfeatures features at each split
• If float then maxfeatures is a fraction and intmaxfeatures nfeatures features are considered at each split
• If "auto" then maxfeatures = sqrt(nfeatures)
• If "sqrt" then maxfeatures = sqrt(nfeatures)
• If "log2" then maxfeatures = log2(nfeatures)
• If None then maxfeatures = nfeatures
Choosing maxfeatures nfeatures leads to a reduction of variance and an increase in bias
Note the search for a split does not stop until at least one valid partition of the node samples is found even if it requires to effectively inspect more than maxfeatures features
verbose int default 0 Enable verbose output If 1 then it prints progress and performance once in a while the more trees the lower the frequency If greater than 1 then it prints progress and performance for every tree
maxleafnodes int or None optional default None Grow trees with maxleafnodes in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then unlimited number of leaf nodes
warmstart bool default False When set to True reuse the solution of the previous call to fit and add more estimators to the ensemble otherwise just erase the previous solution See the Glossary
presort bool or 'auto' optional default 'auto' Whether to presort the data to speed up the finding of best splits in fitting Auto mode by default will use presorting on dense data and

scikitlearn user guide Release 0213

default to normal sorting on sparse data Setting presort to true on sparse data will raise an error

New in version 017 presort parameter

validationfraction float optional default 0.1 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if niternochange is set to an integer

New in version 020

niternochange int default None niternochange is used to decide if early stopping will be used to terminate training when validation score is not improving By default it is set to None to disable early stopping If set to a number it will set aside validationfraction size of the training data as validation and terminate training when validation score is not improving in all of the previous niternochange numbers of iterations The split is stratified

New in version 020

tolfloat optional default 1e4 Tolerance for the early stopping When the loss is not improving by at least tol for niternochange iterations if set to a number the training stops

New in version 020

Attributes

nestimators int The number of estimators as selected by early stopping if niternochange is specified Otherwise it is set to nestimators

New in version 020

featureimportances array shape nfeatures Return the feature importances the higher the more important the feature

oobimprovement array shape nestimators The improvement in loss deviance on the outofbag samples relative to the previous iteration oobimprovement0 is the improvement in loss of the first stage over the init estimator

trainscore array shape nestimators The ith score trainscorei is the deviance loss of the model at iteration i on the inbag sample If subsample 1 this is the deviance on the training data

loss LossFunction The concrete LossFunction object

init_estimator The estimator that provides the initial predictions Set via the init_argument or lossinit_estimator

estimators ndarray of DecisionTreeRegressor shape nestimators lossK The collection of fitted subestimators lossK is 1 for binary classification otherwise nclasses

See also

sklearnensembleHistGradientBoostingClassifier

sklearnDecisionTreeClassifier RandomForestClassifier

AdaBoostClassifier

528 Chapter 3 User Guide

scikitlearn user guide Release 0213

Notes

The features are always randomly permuted at each split Therefore the best found split may vary even with the same training data and maxfeaturesnfeatures if the improvement of the criterion is identical for several splits enumerated during the search of the best split To obtain a deterministic behaviour during fitting randomstate has to be fixed

References

J Friedman Greedy Function Approximation A Gradient Boosting Machine The Annals of Statistics V ol 29 No 5 2001

10 Friedman Stochastic Gradient Boosting 1999

T Hastie R Tibshirani and J Friedman Elements of Statistical Learning Ed 2 Springer 2009

Methods

apply self X Apply trees in the ensemble to X return leaf indices
decisionfunction self X Compute the decision function of X
fitself X y sampleweight monitor Fit the gradient boosting model
getparams self deep Get parameters for this estimator
predict self X Predict class for X
predictlogproba self X Predict class logprobabilities for X
predictproba self X Predict class probabilities for X
score self X y sampleweight Returns the mean accuracy on the given test data and labels
setparams self params Set the parameters of this estimator
stageddecisionfunction self X Compute decision function of Xfor each iteration
stagedpredict self X Predict class at each stage for X
stagedpredictproba self X Predict class probabilities at each stage for X
init self loss'deviance' learningrate01 nestimators100 subsam
ple10 criterion'friedmanmse' minsamplessplit2 minsamplesleaf1
minweightfractionleaf00 maxdepth3 minimpuritydecrease00
minimpuritysplitNone initNone randomstateNone maxfeaturesNone
verbose0 maxleafnodesNone warmstartFalse presort'auto' valida
tionfraction01 niternochangeNone tol00001
applyselfX

Apply trees in the ensemble to X return leaf indices

New in version 017

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Inter
nally its dtype will be converted to dtypenpfloat32 If a sparse matrix is provided
it will be converted to a sparse csrmatrix

Returns

Xleaves arraylike shape nsamples nestimators nclasses For each datapoint x in X
and for each tree in the ensemble return the index of the leaf x ends up in each estimator
33 Model selection and evaluation 529

scikitlearn user guide Release 0213

In the case of binary classification nclasses is 1

decisionfunction selfX

Compute the decision function of X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Inter

nally it will be converted to dtype npfloat32 and if a sparse matrix is provided to

a sparsecsrmatrix

Returns

score array shape nsamples nclasses or nsamples The decision function of the in

put samples which corresponds to the raw values predicted from the trees of the ensemble

The order of the classes corresponds to that in the attribute classes Regression and

binary classification produce an array of shape nsamples

featureimportances

Return the feature importances the higher the more important the feature

Returns

featureimportances array shape nfeatures The values of this array sum to 1 unless

all trees are single node trees consisting of only the root node in which case it will be an

array of zeros

fitselfXysampleweightNone monitorNone

Fit the gradient boosting model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Inter

nally it will be converted to dtype npfloat32 and if a sparse matrix is provided to

a sparsecsrmatrix

yarraylike shape nsamples Target values strings or integers in classification real

numbers in regression For classification labels must correspond to classes

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Splits that would create child nodes with net zero or neg

ative weight are ignored while searching for a split in each node In the case of classifi

cation splits are also ignored if they would result in any single class carrying a negative

weight in either child node

monitor callable optional The monitor is called after each iteration with the current iter

ation a reference to the estimator and the local variables of fitstages as keyword

arguments callable self locals If the callable returns True the fit

ting procedure is stopped The monitor can be used for various things such as computing

heldout estimates early stopping model introspect and snapshoting

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

530 Chapter 3 User Guide

scikitlearn user guide Release 0213

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class for X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

Returns

yarray shape nsamples The predicted values

predictlogproba selfX

Predict class logprobabilities for X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

Returns

parray shape nsamples nclasses The class logprobabilities of the input samples The order of the classes corresponds to that in the attribute classes

Raises

AttributeError If the loss does not support probabilities

predictproba selfX

Predict class probabilities for X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

Returns

parray shape nsamples nclasses The class probabilities of the input samples The order of the classes corresponds to that in the attribute classes

Raises

AttributeError If the loss does not support probabilities

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

33 Model selection and evaluation 531

scikitlearn user guide Release 0213

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

stageddecisionfunction selfX

Compute decision function of Xfor each iteration

This method allows monitoring ie determine error on testing set after each stage

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsrmatrix

Returns

score generator of array shape nsamples k The decision function of the input samples which corresponds to the raw values predicted from the trees of the ensemble The classes corresponds to that in the attribute classes Regression and binary classification are special cases with k 1 otherwise knclasses

stagedpredict selfX

Predict class at each stage for X

This method allows monitoring ie determine error on testing set after each stage

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsrmatrix

Returns

ygenerator of array of shape nsamples The predicted value of the input samples

stagedpredictproba selfX

Predict class probabilities at each stage for X

This method allows monitoring ie determine error on testing set after each stage

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsrmatrix

Returns

ygenerator of array of shape nsamples The predicted value of the input samples

scikitlearn user guide Release 0213

Examples using sklearnensembleGradientBoostingClassifier

- Gradient Boosting regularization
- Early stopping of Gradient Boosting
- Feature transformations with ensembles of trees
- Gradient Boosting OutofBag estimates
- Feature discretization

sklearnensemble GradientBoostingRegressor

classsklearnensemble GradientBoostingRegressor loss'ls' learningrate01

nestimators100 subsam

ple10 criterion'friedmanmse'

minsamplesplit2

minsamplesleaf1

minweightfractionleaf00

maxdepth3

minimpuritydecrease00

minimpuritysplitNone

initNone randomstateNone

maxfeaturesNone alpha09

verbose0 maxleafnodesNone

warmstartFalse pre

sort'auto' validationfraction01

niternochangeNone tol00001

Gradient Boosting for regression

GB builds an additive model in a forward stagewise fashion it allows for the optimization of arbitrary differentiable loss functions In each stage a regression tree is fit on the negative gradient of the given loss function

Read more in the User Guide

Parameters

loss 'ls' 'lad' 'huber' 'quantile' optional default'ls' loss function to be optimized 'ls'

refers to least squares regression 'lad' least absolute deviation is a highly robust loss function solely based on order information of the input variables 'huber' is a combination of the two 'quantile' allows quantile regression use alpha to specify the quantile learningrate float optional default01 learning rate shrinks the contribution of each tree bylearningrate There is a tradeoff between learningrate and nestimators

nestimators int default100 The number of boosting stages to perform Gradient boosting is fairly robust to overfitting so a large number usually results in better performance subsample float optional default10 The fraction of samples to be used for fitting the individual base learners If smaller than 10 this results in Stochastic Gradient Boosting subsample interacts with the parameter nestimators Choosing subsample 10 leads to a reduction of variance and an increase in bias

criterion string optional default"friedmanmse" The function to measure the quality of a split Supported criteria are "friedmanmse" for the mean squared error with improvement score by Friedman "mse" for mean squared error and "mae" for the mean absolute error The default value of "friedmanmse" is generally the best as it can provide a better approximation in some cases

33 Model selection and evaluation 533

scikitlearn user guide Release 0213

New in version 018

minsamplessplit int float optional default2 The minimum number of samples required to split an internal node

- If int then consider minsamplessplit as the minimum number
- If float then minsamplessplit is a fraction and ceilminsamplessplit

nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions

minsamplesleaf int float optional default1 The minimum number of samples required to be at a leaf node A split point at any depth will only be considered if it leaves at least minsamplesleaf training samples in each of the left and right branches This may have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number
- If float then minsamplesleaf is a fraction and ceilminsamplesleaf

nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the sum total of weights of all the input samples required to be at a leaf node Samples have equal weight when sampleweight is not provided

maxdepth integer optional default3 maximum depth of the individual regression estimators The maximum depth limits the number of nodes in the tree Tune this parameter for best performance the best value depends on the interaction of the input variables

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$N_t \cdot N \cdot \text{impurity} - N_t R \cdot N_t \cdot \text{rightimpurity}$$

$$- N_t L \cdot N_t \cdot \text{leftimpurity}$$

where N is the total number of samples N_t is the number of samples at the current node

$N_t L$ is the number of samples in the left child and $N_t R$ is the number of samples in the right child

N , $N_t R$ and $N_t L$ all refer to the weighted sum if sampleweight is passed

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of

minimpuritydecrease in 019 The default value of minimpuritysplit will change from 1e7 to 0 in 023 and it will be removed in 025 Use

minimpuritydecrease instead

init estimator or 'zero' optional defaultNone An estimator object that is used to compute the initial predictions init has to provide fit and predict If 'zero' the initial raw predictions are set to zero By default a DummyEstimator is used predicting either the average target value for loss'ls' or a quantile for the other losses

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance

scikitlearn user guide Release 0213

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

maxfeatures int float string or None optional defaultNone The number of features to consider when looking for the best split

- If int then consider maxfeatures features at each split
- If float then maxfeatures is a fraction and intmaxfeatures nfeatures features are considered at each split
- If “auto” then maxfeaturesnfeatures
- If “sqrt” then maxfeaturesqrtnfeatures
- If “log2” then maxfeatureslog2nfeatures
- If None then maxfeaturesnfeatures

Choosingmaxfeatures nfeatures leads to a reduction of variance and an increase in bias

Note the search for a split does not stop until at least one valid partition of the node samples is found even if it requires to effectively inspect more than maxfeatures features

alpha float default0.9 The alphaquantile of the huber loss function and the quantile loss function Only if loss=huber or loss=quantile

verbose int default 0 Enable verbose output If 1 then it prints progress and performance once in a while the more trees the lower the frequency If greater than 1 then it prints progress and performance for every tree

maxleaffnodes int or None optional defaultNone Grow trees with maxleaffnodes in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then unlimited number of leaf nodes

warmstart bool default False When set to True reuse the solution of the previous call to fit and add more estimators to the ensemble otherwise just erase the previous solution See the Glossary

presort bool or ‘auto’ optional default‘auto’ Whether to presort the data to speed up the finding of best splits in fitting Auto mode by default will use presorting on dense data and default to normal sorting on sparse data Setting presort to true on sparse data will raise an error

New in version 0.17 optional parameter presort

validationfraction float optional default 0.1 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if niternochange is set to an integer

New in version 0.20

niternochange int default None niternochange is used to decide if early stopping will be used to terminate training when validation score is not improving By default it is set to None to disable early stopping If set to a number it will set aside validationfraction size of the training data as validation and terminate training when validation score is not improving in all of the previous niternochange numbers of iterations

New in version 0.20

33 Model selection and evaluation 535

scikitlearn user guide Release 0213

tolfloat optional default 1e4 Tolerance for the early stopping When the loss is not improv
ing by at least tol for niternochange iterations if set to a number the training
stops

New in version 020

Attributes

featureimportances array shape nfeatures Return the feature importances the
higher the more important the feature

oobimprovement array shape nestimators The improvement in loss deviance on
the outofbag samples relative to the previous iteration oobimprovement0 is the
improvement in loss of the first stage over the init estimator

trainscore array shape nestimators The ith score trainscorei is the de
viance loss of the model at iteration ion the inbag sample If subsample 1
this is the deviance on the training data

loss LossFunction The concrete LossFunction object

init estimator The estimator that provides the initial predictions Set via the init argument

orlossinitestimator

estimators array of DecisionTreeRegressor shape nestimators 1 The collection of fitted
subestimators

See also

sklearnensembleHistGradientBoostingRegressor

sklearnmtreeDecisionTreeRegressor RandomForestRegressor

Notes

The features are always randomly permuted at each split Therefore the best found split may vary even with
the same training data and maxfeaturesnfeatures if the improvement of the criterion is identical for
several splits enumerated during the search of the best split To obtain a deterministic behaviour during fitting
randomstate has to be fixed

References

J Friedman Greedy Function Approximation A Gradient Boosting Machine The Annals of Statistics V ol 29
No 5 2001

10 Friedman Stochastic Gradient Boosting 1999

T Hastie R Tibshirani and J Friedman Elements of Statistical Learning Ed 2 Springer 2009

Methods

apply self X Apply trees in the ensemble to X return leaf indices

fitself X y sampleweight monitor Fit the gradient boosting model

getparams self deep Get parameters for this estimator

predict self X Predict regression target for X

Continued on next page

536 Chapter 3 User Guide

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

stagedpredict self X Predict regression target at each stage for X

init self loss'ls' learningrate01 nestimators100 subsample10

criterion'friedmanmse' minsamplessplit2 minsamplesleaf1

minweightfractionleaf00 maxdepth3 minimpuritydecrease00

minimpuritysplitNone initNone randomstateNone maxfeaturesNone al

pha09 verbose0 maxleafnodesNone warmstartFalse presort'auto' valida

tionfraction01 niternochangeNone tol00001

applyselfX

Apply trees in the ensemble to X return leaf indices

New in version 017

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally its dtype will be converted to dtypepenfloat32 If a sparse matrix is provided it will be converted to a sparse csrmatrix

Returns

Xleaves arraylike shape nsamples nestimators For each datapoint x in X and for each tree in the ensemble return the index of the leaf x ends up in each estimator

featureimportances

Return the feature importances the higher the more important the feature

Returns

featureimportances array shape nfeatures The values of this array sum to 1 unless all trees are single node trees consisting of only the root node in which case it will be an array of zeros

fitselfXysampleweightNone monitorNone

Fit the gradient boosting model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtypepenfloat32 and if a sparse matrix is provided to a sparsecsrmatrix

yarraylike shape nsamples Target values strings or integers in classification real

numbers in regression For classification labels must correspond to classes

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Splits that would create child nodes with net zero or negative weight are ignored while searching for a split in each node In the case of classification splits are also ignored if they would result in any single class carrying a negative weight in either child node

monitor callable optional The monitor is called after each iteration with the current iteration a reference to the estimator and the local variables of fitstages as keyword

scikitlearn user guide Release 0213

argumentscallable: self locals If the callable returns True the fitting procedure is stopped The monitor can be used for various things such as computing heldout estimates early stopping model introspect and snapshoting

Returns

self object

getparams selfdeep: True

Get parameters for this estimator

Parameters

deep: boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params: mapping of string to any Parameter names mapped to their values

predictselfX

Predict regression target for X

Parameters

X: arraylike sparse matrix shape (nsamples, nfeatures) The input samples Internally it will be converted to dtype np.float32 and if a sparse matrix is provided to a sparse matrix

Returns

y: array shape (nsamples,) The predicted values

scoreselfX: sampleweight: None

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

\sum^2 and v is the total sum of squares $y_{true} - y_{true\text{mean}}$ \sum^2 The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

X: arraylike shape (nsamples, nfeatures) Test samples For some estimators this may

be a precomputed kernel matrix instead shape (nsamples, nsamplesfitted) where

nsamplesfitted is the number of samples used in the fitting for the estimator

y: arraylike shape (nsamples,) or (nsamples, noutputs) True values for X

sampleweight: arraylike shape (nsamples,) optional Sample weights

Returns

score: float R2 of self.predictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 0.23 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

538 Chapter 3 User Guide

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

stagedpredict selfX

Predict regression target at each stage for X

This method allows monitoring ie determine error on testing set after each stage

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsrmatrix

Returns

ygenerator of array of shape nsamples The predicted value of the input samples

Examples using sklearnensembleGradientBoostingRegressor

- Model Complexity Influence
- Plot individual and voting regression predictions
- Prediction Intervals for Gradient Boosting Regression
- Gradient Boosting regression
- Partial Dependence Plots

333 Model evaluation quantifying the quality of predictions

There are 3 different APIs for evaluating the quality of a model's predictions

•Estimator score method Estimators have a score method providing a default evaluation criterion for the problem they are designed to solve This is not discussed on this page but in each estimator's documentation

•Scoring parameter Modevaluation tools using crossvalidation such asmodelselection crossvalscore andmodelselectionGridSearchCV rely on an internal scoring strategy This

is discussed in the section The scoring parameter defining model evaluation rules

•Metric functions Themetrics module implements functions assessing prediction error for specific purposes These metrics are detailed in sections on Classification metrics Multilabel ranking metrics Regression metrics andClustering metrics

Finally Dummy estimators are useful to get a baseline value of those metrics for random predictions

See also

For "pairwise" metrics between samples and not estimators or predictions see the Pairwise metrics Affinities and Kernels section

33 Model selection and evaluation 539

The scoring parameter defining model evaluation rules

Model selection and evaluation using tools such as `modelselectionGridSearchCV` and

`modelselectioncrossvalscore` take as scoring parameter that controls what metric they ap

ply to the estimators evaluated

Common cases predefined values

For the most common use cases you can designate a scorer object with the scoring parameter the table below

shows all possible values All scorer objects follow the convention that higher return values are better than

lower return values Thus metrics which measure the distance between the model and the data like metrics

`meansquarederror` are available as `negmeansquarederror` which return the negated value of the metric

Scoring Function Comment

Classification

`'accuracy'` `metricsaccuracy`

`'balancedaccuracy'` `metricsbalancedaccuracy`

`'averageprecision'` `metricsaverageprecision`

`'brierscoreloss'` `metricsbrierscoreloss`

`'f1'` `metricsf1score` for binary targets

`'f1micro'` `metricsf1score` microaveraged

`'f1macro'` `metricsf1score` macroaveraged

`'f1weighted'` `metricsf1score` weighted average

`'f1samples'` `metricsf1score` by multilabel sample

`'neglogloss'` `metricslogloss` requires `predict_proba` support

`'precision'` etc `metricsprecision` suffixes apply as with `'f1'`

`'recall'` etc `metricsrecall` suffixes apply as with `'f1'`

`'jaccard'` etc `metricsjaccard` suffixes apply as with `'f1'`

`'rocauc'` `metricsrocauc`

Clustering

`'adjustedmutualinfoscore'` `metricsadjustedmutualinfoscore`

`'adjustedrandscore'` `metricsadjustedrandscore`

`'completenessscore'` `metricscompletenessscore`

`'fowlkesmallowsscore'` `metricsfowlkesmallowsscore`

`'homogeneityscore'` `metricshomogeneityscore`

`'mutualinfoscore'` `metricsmutualinfoscore`

`'normalizedmutualinfoscore'` `metricsnormalizedmutualinfoscore`

`'vmeasurescore'` `metricsvmeasurescore`

Regression

`'explainedvariance'` `metricsexplainedvariancescore`

`'maxerror'` `metricsmaxerror`

`'negmeanabsoluteerror'` `metricsmeanabsoluteerror`

`'negmeansquarederror'` `metricsmeansquarederror`

`'negmeansquaredlogerror'` `metricsmeansquaredlogerror`

`'negmedianabsoluteerror'` `metricsmedianabsoluteerror`

`'r2'` `metricsr2score`

Usage examples

`from sklearn import svm datasets`

`from sklearnmodelselection import crossvalscore`

`iris datasetsloadiris`

scikitlearn user guide Release 0213

X y irisdata iristarget

clf svmSVCgamma scale randomstate0

crossvalscoreclf X y scoringrecallmacro

cv5

array096 096 096 093 1

model svmSVC

crossvalscoremodel X y cv5 scoringwrongchoice

Traceback most recent call last

ValueError wrongchoice is not a valid scoring value Use sortedsklearnmetrics

→SCORERSkeys to get valid options

Note The values listed by the ValueError exception correspond to the functions measuring prediction accuracy described in the following sections The scorer objects for those functions are stored in the dictionary sklearn metricsSCORERS

Defining your scoring strategy from metric functions

The module sklearnmetrics also exposes a set of simple functions measuring a prediction error given ground truth and prediction

- functions ending with score return a value to maximize the higher the better
- functions ending with error orloss return a value to minimize the lower the better When converting into a scorer object using makescorer set thegreaterisbetter parameter to False True by default see the parameter description below

Metrics available for various machine learning tasks are detailed in sections below

Many metrics are not given names to be used as scoring values sometimes because they require additional parameters such as fbeta score In such cases you need to generate an appropriate scoring object The simplest way to generate a callable object for scoring is by using makescorer That function converts metrics into callables that can be used for model evaluation

One typical use case is to wrap an existing metric function from the library with nondefault values for its parameters such as the beta parameter for the fbeta score function

```
from sklearnmetrics import fbeta score makescorer
ftwoscorer makescorerfbeta score beta2
from sklearnmodelselection import GridSearchCV
from sklearnsvm import LinearSVC
grid GridSearchCVLinearSVC paramgridC 1 10
scoringftwoscorer cv5
```

The second use case is to build a completely custom scorer object from a simple python function using makescorer which can take several parameters

- the python function you want to use mycustomlossfunc in the example below
- whether the python function returns a score greaterisbetterTrue the default or a loss greaterisbetterFalse If a loss the output of the python function is negated by the scorer object conforming to the cross validation convention that scorers return higher values for better models
- for classification metrics only whether the python function you provided requires continuous decision certain ties needsthresholdTrue The default value is False
- any additional parameters such as beta orlabels inf1score

33 Model selection and evaluation 541

scikitlearn user guide Release 0213

Here is an example of building custom scorers and of using the greaterisbetter parameter

```
import numpy as np
def mycustomlossfuncytrue ypred
diff npabsytrue ypredmax
return nplog1pdiff
```

score will negate the return value of mycustomlossfunc
which will be nplog2 0693 given the values for X
and y defined below

```
score makescorermycustomlossfunc greaterisbetter False
X 1 1
y 0 1
from sklearndummy import DummyClassifier
clf DummyClassifierstrategymostfrequent randomstate0
clf clffitX y
mycustomlossfuncclfpredictX y
069
scoreclf X y
069
```

Implementing your own scoring object

You can generate even more flexible model scorers by constructing your own scoring object from scratch without using
themakescorer factory For a callable to be a scorer it needs to meet the protocol specified by the following two
rules

- It can be called with parameters estimator X y whereestimator is the model that should be
evaluatedXis validation data and yis the ground truth target for Xin the supervised case or None in the
unsupervised case
- It returns a floating point number that quantifies the estimator prediction quality on X with reference to y

Again by convention higher numbers are better so if your scorer returns loss that value should be negated

Note Using custom scorers in functions where njobs 1

While defining the custom scoring function alongside the calling function should work out of the box with the default
joblib backend loky importing it from another module will be a more robust approach and work independently of
the joblib backend

For example to use njobs greater than 1 in the example below customscoringfunction function is saved
in a usercreated module customscorermodulepy and imported

```
from customscorermodule import customscoringfunction
crossvalscoremodel
Xtrain
ytrain
scoringmakescorercustomscoringfunction greaterisbetter False
cv5
njobs1
```

scikitlearn user guide Release 0213

Using multiple metric evaluation

Scikitlearn also permits evaluation of multiple metrics in GridSearchCV RandomizedSearchCV and crossvalidate

There are two ways to specify multiple scoring metrics for the scoring parameter

- As an iterable of string metrics

scoring accuracy precision

- As a dict mapping the scorer name to the scoring function

from sklearn.metrics import accuracyscore

from sklearn.metrics import makescorer

scoring accuracy makescoreraccuracyscore

prec precision

Note that the dict values can either be scorer functions or one of the predefined metric strings

Currently only those scorer functions that return a single score can be passed inside the dict Scorer functions that

return multiple values are not permitted and will require a wrapper to return a single metric

from sklearn.model_selection import crossvalidate

from sklearn.metrics import confusionmatrix

A sample toy binary classification dataset

X y datasets.make_classification(n_classes=2, random_state=0)

svm = LinearSVC(random_state=0)

def tnytrue ypred return confusionmatrix(ytrue, ypred)[0][0]

def fpytrue ypred return confusionmatrix(ytrue, ypred)[0][1]

def fnytrue ypred return confusionmatrix(ytrue, ypred)[1][0]

def tpytrue ypred return confusionmatrix(ytrue, ypred)[1][1]

scoring = {'tp': makescorer(tp), 'tn': makescorer(tn),

'fp': makescorer(fp), 'fn': makescorer(fn)}

cv_results = cross_validate(svm_fit, X, y, scoring=scoring, cv=5)

scoring = scoring[0]

Getting the test set true positive scores

print cv_results['test_tp']

10 9 8 7 8

Getting the test set false negative scores

print cv_results['test_fn']

0 1 2 3 2

Classification metrics

The sklearn.metrics module implements several loss score and utility functions to measure classification per-

formance. Some metrics might require probability estimates of the positive class confidence values or binary deci-

sions values. Most implementations allow each sample to provide a weighted contribution to the overall score through

the sample_weight parameter.

Some of these are restricted to the binary classification case.

precision_recall_curve(ytrue, probascore) Compute precision-recall pairs for different probability

thresholds

roc_curve(ytrue, yscore, poslabel) Compute Receiver operating characteristic (ROC)

balanced_accuracy_score(ytrue, ypred) Compute the balanced accuracy

33 Model selection and evaluation 543

scikitlearn user guide Release 0213

Others also work in the multiclass case

cohenkappascore y1 y2 labels weights Cohen’s kappa a statistic that measures interannotator agreement

confusionmatrix ytrue ypred labels Compute confusion matrix to evaluate the accuracy of a classification

hinge loss ytrue preddecision labels Average hinge loss nonregularized

matthewscore ytrue ypred Compute the Matthews correlation coefficient MCC

Some also work in the multilabel case

accuracy ytrue ypred normalize Accuracy classification score

classificationreport ytrue ypred Build a text report showing the main classification metrics

f1score ytrue ypred labels Compute the F1 score also known as balanced Fscore or Fmeasure

fbetascore ytrue ypred beta labels Compute the Fbeta score

hammingloss ytrue ypred labels Compute the average Hamming loss

jaccardscore ytrue ypred labels Jaccard similarity coefficient score

logloss ytrue ypred eps normalize Log loss aka logistic loss or crossentropy loss

multilabelconfusionmatrix ytrue ypred Compute a confusion matrix for each class or sample

precisionrecallfscoresupport ytrue ypred Compute precision recall Fmeasure and support for each class

precisionscore ytrue ypred labels Compute the precision

recallscore ytrue ypred labels Compute the recall

zeroone loss ytrue ypred normalize Zeroone classification loss

And some work with binary and multilabel but not multiclass problems

averageprecisionscore ytrue yscore Compute average precision AP from prediction scores

rocaucscore ytrue yscore average Compute Area Under the Receiver Operating Characteristic Curve ROC AUC from prediction scores

In the following subsections we will describe each of those functions preceded by some notes on common API (Application Programming Interface) metric definition

From binary to multiclass and multilabel

Some metrics are essentially defined for binary classification tasks eg f1score rocaucscore In these

cases by default only the positive label is evaluated assuming by default that the positive class is labelled 1 though this may be configurable through the poslabel parameter

In extending a binary metric to multiclass or multilabel problems the data is treated as a collection of binary problems one for each class There are then a number of ways to average binary metric calculations across the set of classes each of which may be useful in some scenario Where available you should select among these using the average parameter

•macro simply calculates the mean of the binary metrics giving equal weight to each class In problems where infrequent classes are nonetheless important macroaveraging may be a means of highlighting their performance On the other hand the assumption that all classes are equally important is often untrue such that macroaveraging will overemphasize the typically low performance on an infrequent class

544 Chapter 3 User Guide

scikitlearn user guide Release 0213

- weighted accounts for class imbalance by computing the average of binary metrics in which each class’s score is weighted by its presence in the true data sample
 - micro gives each sampleclass pair an equal contribution to the overall metric except as a result of sample weight Rather than summing the metric per class this sums the dividends and divisors that make up the perclass metrics to calculate an overall quotient Microaveraging may be preferred in multilabel settings including multiclass classification where a majority class is to be ignored
 - samples applies only to multilabel problems It does not calculate a perclass measure instead calculat ing the metric over the true and predicted classes for each sample in the evaluation data and returning their sampleweight weighted average
 - Selecting averageNone will return an array with the score for each class
- While multiclass data is provided to the metric like binary targets as an array of class labels multilabel data is specified as an indicator matrix in which cell i j has value 1 if sample ihas labeljand value 0 otherwise

Accuracy score

Theaccuracyscore function computes the accuracy either the fraction default or the count normalizeFalse of correct predictions

In multilabel classification the function returns the subset accuracy If the entire set of predicted labels for a sample strictly match with the true set of labels then the subset accuracy is 10 otherwise it is 00

If \hat{y}_i is the predicted value of the i th sample and y_i is the corresponding true value then the fraction of correct predictions over n samples is defined as

$$\text{accuracy} = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{\hat{y}_i = y_i}$$

where $\mathbb{1}$ is the indicator function

```
import numpy as np
from sklearn.metrics import accuracy_score

ypred = [0, 2, 1, 3]
ytrue = [0, 1, 2, 3]
accuracy_score(ytrue, ypred)
0.5
accuracy_score(ytrue, ypred, normalize=False)
2
```

In the multilabel case with binary label indicators

```
accuracy_score(np.array([0, 1, 1, 1]), np.ones(2, 2))
0.5
```

- Example
- See Test with permutations the significance of a classification score for an example of accuracy score usage using permutations of the dataset
- 33 Model selection and evaluation 545

Balanced accuracy score

The `balancedaccruracyscore` function computes the balanced accuracy which avoids inflated performance estimates on imbalanced datasets It is the macroaverage of recall scores per class or equivalently raw accuracy where each sample is weighted according to the inverse prevalence of its true class Thus for balanced datasets the score is equal to accuracy

In the binary case balanced accuracy is equal to the arithmetic mean of sensitivity true positive rate and specificity true negative rate or the area under the ROC curve with binary predictions rather than scores

If the classifier performs equally well on either class this term reduces to the conventional accuracy ie the number of correct predictions divided by the total number of predictions

In contrast if the conventional accuracy is above chance only because the classifier takes advantage of an imbalanced test set then the balanced accuracy as appropriate will drop to 1

0000000000

The score ranges from 0 to 1 or when adjustedTrue is used it rescaled to the range 1

1-0000000000 to 1 inclusive

with performance at random scoring 0

If y_i is the true value of the i th sample and \hat{y}_i is the corresponding sample weight then we adjust the sample weight to

0000000000

$\hat{y}_i = 1 - y_i$

where 1_{ij} is the indicator function Given predicted \hat{y}_i for sample i balanced accuracy is defined as

balanced accuracy $\hat{y}_i = 1 - \sum_{j=1}^n \hat{y}_{ij}$

00000000 00

With `adjustedTrue` balanced accuracy reports the relative increase from balanced accuracy $\hat{y}_i = 0$

1

0000000000 In the binary case this is also known as Youden’s J statistic or informedness

Note The multiclass definition here seems the most reasonable extension of the metric used in binary classification though there is no certain consensus in the literature

- Our definition Mosley2013 Kelleher2015 and Guyon2015 where Guyon2015 adopt the adjusted version to ensure that random predictions have a score of 0 and perfect predictions have a score of 1
- Class balanced accuracy as described in Mosley2013 the minimum between the precision and the recall for each class is computed Those values are then averaged over the total number of classes to get the balanced accuracy
- Balanced Accuracy as described in Urbanowicz2015 the average of sensitivity and specificity is computed for each class and then averaged over total number of classes

References

Cohen’s kappa

The function `cohenkappascore` computes Cohen’s kappa statistic This measure is intended to compare label ings by different human annotators not a classifier versus a ground truth

The kappa score see `docstring` is a number between -1 and 1 Scores above 0.8 are generally considered good agreement zero or lower means no agreement practically random labels

scikitlearn user guide Release 0213

Kappa scores can be computed for binary or multiclass problems but not for multilabel problems except by manually computing a perlabel score and not for more than two annotators

```
from sklearn.metrics import cohenkappascore
```

```
ytrue 2 0 2 2 0 1
```

```
ypred 0 0 2 2 0 2
```

```
cohenkappascore(ytrue, ypred)
```

```
0.4285714285714286
```

Confusion matrix

The confusion matrix function evaluates classification accuracy by computing the confusion matrix with each row corresponding to the true class https://en.wikipedia.org/wiki/Confusion_matrix. Wikipedia and other references may use different convention for axes

By definition entry C_{ij} in a confusion matrix is the number of observations actually in group j but predicted to be in group i . Here is an example

```
from sklearn.metrics import confusion_matrix
```

```
ytrue 2 0 2 2 0 1
```

```
ypred 0 0 2 2 0 2
```

```
confusion_matrix(ytrue, ypred)
```

```
array([[2, 0, 0]
```

```
       [0, 0, 1]
```

```
       [1, 0, 2]])
```

Here is a visual representation of such a confusion matrix this figure comes from the Confusion matrix example

For binary problems we can

get counts of true negatives false positives false negatives and true positives as follows

```
ytrue 0 0 0 1 1 1 1 1
```

```
ypred 0 1 0 1 0 1 0 1
```

```
tn fp fn tp confusion_matrix(ytrue, ypred).ravel()
```

```
33 Model selection and evaluation 547
```

tn fp fn tp

2 1 2 3

Example

- See Confusion matrix for an example of using a confusion matrix to evaluate classifier output quality
- See Recognizing handwritten digits for an example of using a confusion matrix to classify handwritten digits
- See Classification of text documents using sparse features for an example of using a confusion matrix to classify text documents

Classification report

Theclassificationreport function builds a text report showing the main classification metrics Here is a small example with custom targetnames and inferred labels

```
from sklearn.metrics import classification_report
ytrue 0 1 2 2 0
ypred 0 0 2 1 0
targetnames class 0 class 1 class 2
printclassification_report(ytrue, ypred, targetnames)
precision recall f1score support
class 0 0.67 1.00 0.80 2
class 1 0.00 0.00 0.00 1
class 2 1.00 0.50 0.67 2
accuracy 0.60 5
macro avg 0.56 0.50 0.49 5
weighted avg 0.67 0.60 0.59 5
```

Example

- See Recognizing handwritten digits for an example of classification report usage for handwritten digits
- See Classification of text documents using sparse features for an example of classification report usage for text documents
- See Parameter estimation using grid search with crossvalidation for an example of classification report usage for grid search with nested crossvalidation

Hamming loss

Thehammingloss computes the average Hamming loss or Hamming distance between two sets of samples If is the predicted value for the th label of a given sample is the corresponding true value and labels is the number of classes or labels then the Hamming loss between two samples is defined as

$$\frac{1}{\text{labels}} \sum_{i=1}^{\text{labels}} \frac{|y_i - \hat{y}_i|}{\text{labels}}$$

scikitlearn user guide Release 0213

where $1_{\{i\}}$ is the indicator function

```
from sklearn.metrics import hammingloss
ypred = [1, 2, 3, 4]
ytrue = [2, 2, 3, 4]
hammingloss(ytrue, ypred)
```

0.25

In the multilabel case with binary label indicators

```
hammingloss(np.array([0, 1, 1, 1]), np.zeros(2))
```

0.75

Note In multiclass classification the Hamming loss corresponds to the Hamming distance between y_{true} and y_{pred} which is similar to the Zero one loss function. However while zeroone loss penalizes prediction sets that do not strictly match true sets the Hamming loss penalizes individual labels. Thus the Hamming loss upper bounded by the zeroone loss is always between zero and one inclusive and predicting a proper subset or superset of the true labels will give a Hamming loss between zero and one exclusive.

Precision recall and Fmeasures

Intuitively precision is the ability of the classifier not to label as positive a sample that is negative and recall is the ability of the classifier to find all the positive samples.

The Fmeasure $\frac{2pr}{p+r}$ and $\frac{2}{\frac{1}{p} + \frac{1}{r}}$ measures can be interpreted as a weighted harmonic mean of the precision and recall. A $\frac{2pr}{p+r}$ measure reaches its best value at 1 and its worst score at 0. With $\frac{1}{\frac{1}{p} + \frac{1}{r}}$ and $\frac{2pr}{p+r}$ are equivalent and the recall and the precision are equally important.

The `precisionrecallcurve` computes a precision-recall curve from the ground truth label and a score given by the classifier by varying a decision threshold.

The `averageprecisionscore` function computes the average precision AP from prediction scores. The value is between 0 and 1 and higher is better. AP is defined as

$$AP = \frac{\sum_{n=1}^N p_n - p_N}{N}$$

where p_n and p_N are the precision and recall at the n th threshold. With random predictions the AP is the fraction of positive samples.

References Manning2008 and Everingham2010 present alternative variants of AP that interpolate the precision-recall curve. Currently `averageprecisionscore` does not implement any interpolated variant. References

Davis2006 and Flach2015 describe why a linear interpolation of points on the precision-recall curve provides an overly optimistic measure of classifier performance. This linear interpolation is used when computing area under the curve with the trapezoidal rule in `auc`.

Several functions allow you to analyze the precision recall and Fmeasures score.

`averageprecisionscore(ytrue, yscore)` Compute average precision AP from prediction scores

`f1score(ytrue, ypred, labels)` Compute the F1 score also known as balanced Fscore or

Fmeasure

`fbetascore(ytrue, ypred, beta, labels)` Compute the Fbeta score

`precisionrecallcurve(ytrue, probaspred)` Compute precision-recall pairs for different probability thresholds

Continued on next page

precisionrecallfscoresupport ytrue
ypredCompute precision recall Fmeasure and support for each
class

precisionscore ytrue ypred labels Compute the precision

recallscore ytrue ypred labels Compute the recall

Note that the precisionrecallcurve function is restricted to the binary case The
averageprecisionscore function works only in binary classification and multilabel indicator format

Examples

- See Classification of text documents using sparse features for an example of f1score usage to classify text documents
- See Parameter estimation using grid search with crossvalidation for an example of precisionscore andrecallscore usage to estimate parameters using grid search with nested crossvalidation
- See PrecisionRecall for an example of precisionrecallcurve usage to evaluate classifier output quality

References

Binary classification

In a binary classification task the terms “positive” and “negative” refer to the classifier’s prediction and the terms “true” and “false” refer to whether that prediction corresponds to the external judgment sometimes known as the “observation” Given these definitions we can formulate the following table

Actual class observation

Predicted class expectation tp true positive Correct result fp false positive Unexpected result

fn false negative Missing result tn true negative Correct absence of result

In this context we can define the notions of precision recall and Fmeasure

precision $\frac{tp}{tp+fp}$

$\frac{precision}{recall}$

recall $\frac{tp}{tp+fn}$

$\frac{precision \times recall}{2 \times precision + 2 \times recall}$

$\frac{1}{\frac{1}{2 \times precision} + \frac{1}{2 \times recall}}$

$\frac{2 \times precision \times recall}{precision + recall}$

Here are some small examples in binary classification

from sklearn import metrics

ypred 0 1 0 0

ytrue 0 1 0 1

metricsprecisionscoreytrue ypred

10

metricsrecallscoreytrue ypred

05

```
scikitlearn user guide Release 0213
metricsf1scoreytrue ypred
066
metricsfbetascoreytrue ypred beta05
083
metricsfbetascoreytrue ypred beta1
066
metricsfbetascoreytrue ypred beta2
055
metricsprecisionrecallfscoresupportytrue ypred beta05
array066 1 array1 05 array071 083 array2
↪2
import numpy as np
from sklearnmetrics import precisionrecallcurve
from sklearnmetrics import averageprecisionscore
ytrue nparray0 0 1 1
yscores nparray01 04 035 08
precision recall threshold precisionrecallcurveytrue yscores
precision
array066 05 1 1
recall
array1 05 05 0
threshold
array035 04 08
averageprecisionscoreytrue yscores
083
```

Multiclass and multilabel classification

In multiclass and multilabel classification task the notions of precision recall and Fmeasures can be applied to each label independently There are a few ways to combine results across labels specified by the average argument to the averageprecisionscore multilabel only f1score fbetascore precisionrecallfscoresupport precisionscore andrecallscore functions as described above Note that if all labels are included “micro”averaging in a multiclass setting will produce precision recall and that are all identical to accuracy Also note that “weighted” averaging may produce an Fscore that is not between precision and recall

To make this more explicit consider the following notation

- the set of predicted pairs
 - the set of true pairs
 - the set of labels
 - the set of samples
 - the subset of with sample ie ‘ \in ’
 - the subset of with label
 - similarly and are subsets of
 - \cap
- for some sets and
- \cap

Conventions vary on handling \emptyset this implementation uses $\emptyset = 0$ and similar for

```

0000 0
1 020000x000
02000000
Then the metrics are defined as
average Precision Recall Fbeta
micro 000 000 0000
samples1
0Σ
0∈0000001
0Σ
0∈0000001
0Σ
0∈0000000
macro1
0Σ
0∈0000001
0Σ
0∈0000001
0Σ
0∈0000000
weighted1Σ
0∈0000Σ
0∈0000000001Σ
0∈0000Σ
0∈0000000001Σ
0∈0000Σ
0∈000000000
None(000000∈0)(000000∈0)(000000∈0)

```

```
033 metricsf1scoreytrue ypred averageweighted
026 metricsfbetascoreytrue ypred averagemacro beta05
023 metricsprecisionrecallf1scoresupportytrue ypred beta05 average None
```

Similarly labels not present in the data sample may be accounted for in macroaveraging

```
metricsprecisionscoreytrue ypred labels0 1 2 3 averagemacro
```

In the binary case
552 Chapter 3 User Guide

scikitlearn user guide Release 0213

```
import numpy as np
from sklearn.metrics import jaccardscore
ytrue = np.array([1, 1, 0])
ypred = np.array([1, 1, 1])
jaccardscore(ytrue, ypred)
0.6666666666666667
```

In the multilabel case with binary label indicators

```
jaccardscore(ytrue, ypred, average='samples')
0.5833333333333333
jaccardscore(ytrue, ypred, average='macro')
0.6666666666666667
```

```
jaccardscore(ytrue, ypred, average=None)
array([0.5, 0.5, 1.])
```

Multiclass problems are binarized and treated like the corresponding multilabel problem

```
ypred = [0, 2, 1, 2]
ytrue = [0, 1, 2, 2]
jaccardscore(ytrue, ypred, average=None)
array([1., 0., 0.33333333, 0.])
```

```
jaccardscore(ytrue, ypred, average='macro')
0.4444444444444444
jaccardscore(ytrue, ypred, average='micro')
0.3333333333333333
```

Hinge loss
The hinge loss function computes the average distance between the model and the data using hinge loss a one sided metric that considers only prediction errors Hinge loss is used in maximal margin classifiers such as support vector machines

If the labels are encoded with 1 and -1 y is the true value and \hat{y} is the predicted decisions as output by decision function then the hinge loss is defined as

$$L_{\text{Hinge}}(y, \hat{y}) = \max(1 - y\hat{y}, 0)$$

If there are more than two labels hinge loss uses a multiclass variant due to Crammer & Singer Here is the paper describing it

If \hat{y}_i is the predicted decision for true label and $\max_{j \neq i} \hat{y}_j$ is the maximum of the predicted decisions for all other labels where predicted decisions are output by decision function then multiclass hinge loss is defined by

$$L_{\text{Hinge}}(y, \hat{y}) = \max(1 - y\hat{y}, \max_{j \neq y} (y\hat{y}_j - \hat{y}_j))$$

Here a small example demonstrating the use of the hinge loss function with a svm classifier in a binary class problem

```
from sklearn import svm
from sklearn.metrics import hinge_loss
X = [0, 1]
y = [1, 1]
```

```
scikitlearn user guide Release 0213
est svmLinearSVCrandomstate0
estfitX y
LinearSVCC10 classweightNone dualTrue fitinterceptTrue
interceptscaling1 losssquaredhinge maxiter1000
multiclassovr penaltyl2 randomstate0 tol00001
verbose0
preddecision estdecisionfunction2 3 05
preddecision
array218 236 009
hingeloss1 1 1 preddecision
03
```

Here is an example demonstrating the use of the hingeloss function with a svm classifier in a multiclass problem

```
X nparray0 1 2 3
Y nparray0 1 2 3
labels nparray0 1 2 3
est svmLinearSVC
estfitX Y
LinearSVCC10 classweightNone dualTrue fitinterceptTrue
interceptscaling1 losssquaredhinge maxiter1000
multiclassovr penaltyl2 randomstateNone tol00001
verbose0
preddecision estdecisionfunction1 2 3
ytrue 0 2 3
hingelossytrue preddecision labels
056
```

Log loss
Log loss also called logistic regression loss or crossentropy loss is defined on probability estimates It is commonly used in multinomial logistic regression and neural networks as well as in some variants of expectationmaximization and can be used to evaluate the probability outputs predictproba of a classifier instead of its discrete predictions

For binary classification with a true label $y \in \{0,1\}$ and a probability estimate $\hat{p} = \text{Pr}[Y=1]$ the log loss per sample is the negative loglikelihood of the classifier given the true label

$$-\log \hat{p}^y - \log \text{Pr}[Y=y] = -\log \hat{p}^1 - (1-\hat{p}) \log 1 - (1-\hat{p}) \log 1 - \hat{p}$$

This extends to the multiclass case as follows Let the true labels for a set of samples be encoded as a 1ofK binary indicator matrix Y ie $Y_{ij} = 1$ if sample i has label j taken from a set of K labels Let \hat{p} be a matrix of probability estimates with $\hat{p}_{ij} = \text{Pr}[Y_{ij}=1]$ Then the log loss of the whole set is

$$-\log \hat{p}^y - \log \text{Pr}[Y=y] = -1 - \sum_{j=1}^K \hat{p}_{ij} \log \hat{p}_{ij}$$

To see how this generalizes the binary log loss given above note that in the binary case $\hat{p}_{i0} = 1 - \hat{p}_{i1}$ and $\hat{p}_{i0} \log \hat{p}_{i0} = (1 - \hat{p}_{i1}) \log (1 - \hat{p}_{i1})$ so expanding the inner sum over $\hat{p}_{ij} \in \{0,1\}$ gives the binary log loss

The logloss function computes log loss given a list of groundtruth labels and a probability matrix as returned by an estimator's predictproba method

```
from sklearn.metrics import logloss
```

```
ytrue 0 0 1 1
```


scikitlearn user guide Release 0213

ypred 9 1 8 2 3 7 0 1 99

loglossytrue ypred

01738

The first9 1 in ypred denotes 90 probability that the first sample has label 0 The log loss is nonnegative Matthews correlation coefficient

The matthewscore function computes the Matthew’s correlation coefficient MCC for binary classes

Quoting Wikipedia

“The Matthews correlation coefficient is used in machine learning as a measure of the quality of binary twoclass classifications It takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes The MCC is in essence a correlation coefficient value between 1 and 1 A coefficient of 1 represents a perfect prediction 0 an average random prediction and 1 an inverse prediction The statistic is also known as the phi coefficient”

In the binary twoclass case TP and FN are respectively the number of true positives true negatives false positives and false negatives the MCC is defined as

$$\frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

In the multiclass case the Matthews correlation coefficient can be defined in terms of a confusionmatrix C for

C classes To simplify the definition consider the following intermediate variables

• $\sum_i C_{ii}$

$\sum_i C_{ii}$ the number of times class i truly occurred

• $\sum_j C_{ij}$

$\sum_j C_{ij}$ the number of times class i was predicted

• $\sum_i C_{ii}$

$\sum_i C_{ii}$ the total number of samples correctly predicted

• $\sum_i \sum_j C_{ij}$

$\sum_i \sum_j C_{ij}$ the total number of samples

Then the multiclass MCC is defined as

$$\frac{\sum_i C_{ii} \times \sqrt{\sum_j C_{ij} \times \sum_k C_{ik}} - \sqrt{(\sum_i C_{ii})^2 \times \sum_j \sum_k C_{jk}}}{\sqrt{(\sum_i \sum_j C_{ij})^2 - \sum_i \sum_j C_{ij}^2}}$$

$$\frac{\sum_i C_{ii} \times \sqrt{\sum_j C_{ij} \times \sum_k C_{ik}} - \sqrt{(\sum_i C_{ii})^2 \times \sum_j \sum_k C_{jk}}}{\sqrt{(\sum_i \sum_j C_{ij})^2 - \sum_i \sum_j C_{ij}^2}}$$

$$\frac{\sum_i C_{ii} \times \sqrt{\sum_j C_{ij} \times \sum_k C_{ik}} - \sqrt{(\sum_i C_{ii})^2 \times \sum_j \sum_k C_{jk}}}{\sqrt{(\sum_i \sum_j C_{ij})^2 - \sum_i \sum_j C_{ij}^2}}$$

$$\frac{\sum_i C_{ii} \times \sqrt{\sum_j C_{ij} \times \sum_k C_{ik}} - \sqrt{(\sum_i C_{ii})^2 \times \sum_j \sum_k C_{jk}}}{\sqrt{(\sum_i \sum_j C_{ij})^2 - \sum_i \sum_j C_{ij}^2}}$$

$$\frac{\sum_i C_{ii} \times \sqrt{\sum_j C_{ij} \times \sum_k C_{ik}} - \sqrt{(\sum_i C_{ii})^2 \times \sum_j \sum_k C_{jk}}}{\sqrt{(\sum_i \sum_j C_{ij})^2 - \sum_i \sum_j C_{ij}^2}}$$

$$\frac{\sum_i C_{ii} \times \sqrt{\sum_j C_{ij} \times \sum_k C_{ik}} - \sqrt{(\sum_i C_{ii})^2 \times \sum_j \sum_k C_{jk}}}{\sqrt{(\sum_i \sum_j C_{ij})^2 - \sum_i \sum_j C_{ij}^2}}$$

$$\frac{\sum_i C_{ii} \times \sqrt{\sum_j C_{ij} \times \sum_k C_{ik}} - \sqrt{(\sum_i C_{ii})^2 \times \sum_j \sum_k C_{jk}}}{\sqrt{(\sum_i \sum_j C_{ij})^2 - \sum_i \sum_j C_{ij}^2}}$$

$$\frac{\sum_i C_{ii} \times \sqrt{\sum_j C_{ij} \times \sum_k C_{ik}} - \sqrt{(\sum_i C_{ii})^2 \times \sum_j \sum_k C_{jk}}}{\sqrt{(\sum_i \sum_j C_{ij})^2 - \sum_i \sum_j C_{ij}^2}}$$

When there are more than two labels the value of the MCC will no longer range between 1 and 1 Instead the minimum value will be somewhere between 1 and 0 depending on the number and distribution of ground true labels The maximum value is always 1

Here is a small example illustrating the usage of the matthewscore function

from sklearn.metrics import matthewscore

ytrue 1 1 1 1

ypred 1 1 1 1

matthewscore(ytrue, ypred)

0.33

Multilabel confusion matrix

The multilabelconfusionmatrix function computes classwise default or samplewise sample

wise True multilabel confusion matrix to evaluate the accuracy of a classification multilabelconfusionmatrix

33 Model selection and evaluation 555

scikitlearn user guide Release 0213

also treats multiclass data as if it were multilabel as this is a transformation commonly applied to evaluate multiclass problems with binary classification metrics such as precision recall etc

When calculating classwise multilabel confusion matrix the count of true negatives for class 0 is 00 false negatives is 010 true positives is 011and false positives is 01

Here is an example demonstrating the use of the multilabelconfusionmatrix function with multilabel

indicator matrix input

```
import numpy as np
from sklearn.metrics import multilabelconfusionmatrix
```

```
ytrue = np.array([0 1
```

```
0 1 0
```

```
ypred = np.array([0 0
```

```
0 1 1
```

```
multilabelconfusionmatrix(ytrue, ypred)
```

```
array([[0 1
```

```
0 1
```

```
1 0
```

```
0 1
```

```
0 1
```

```
1 0
```

Or a confusion matrix can be constructed for each sample's labels

```
multilabelconfusionmatrix(ytrue, ypred, samplewise=True)
```

```
array([[0
```

```
1 1
```

```
1 1
```

```
0 1
```

Here is an example demonstrating the use of the multilabelconfusionmatrix function with multiclass

input

```
ytrue = ['cat', 'ant', 'cat', 'cat', 'ant', 'bird']
```

```
ypred = ['ant', 'ant', 'cat', 'cat', 'ant', 'cat']
```

```
multilabelconfusionmatrix(ytrue, ypred)
```

```
labels = ['ant', 'bird', 'cat']
```

```
array([[3 1
```

```
0 2
```

```
5 0
```

```
1 0
```

```
2 1
```

```
1 2
```

Here are some examples demonstrating the use of the multilabelconfusionmatrix function to calculate

recall or sensitivity specificity fall out and miss rate for each class in a problem with multilabel indicator matrix

input

```
ytrue = np.array([0 0 1
```

```
0 1 0
```

```
1 1 0
```

scikitlearn user guide Release 0213

```
ypred nparray0 1 0
0 0 1
1 1 0
mcm multilabelconfusionmatrixxytrue ypred
tn mcm 0 0
tp mcm 1 1
fn mcm 1 0
fp mcm 0 1
tp tp fn
array1 05 0
```

Calculating specificity also called the true negative rate for each class

```
tn tn fp
array1 0 05
```

Calculating fall out also called the false positive rate for each class

```
fp fp tn
array0 1 05
```

Calculating miss rate also called the false negative rate for each class

```
fn fn tp
array0 05 1
```

Receiver operating characteristic ROC

The function roccurve computes the receiver operating characteristic curve or ROC curve Quoting Wikipedia

“A receiver operating characteristic ROC or simply ROC curve is a graphical plot which illustrates the performance of a binary classifier system as its discrimination threshold is varied It is created by plotting the fraction of true positives out of the positives TPR true positive rate vs the fraction of false positives out of the negatives FPR false positive rate at various threshold settings TPR is also known as sensitivity and FPR is one minus the specificity or true negative rate”

This function requires the true binary value and the target scores which can either be probability estimates of the positive class confidence values or binary decisions Here is a small example of how to use the roccurve function

```
import numpy as np
from sklearnmetrics import roccurve
y nparray1 1 2 2
scores nparray01 04 035 08
fpr tpr thresholds roccurve scores poslabel2
fpr
array0 0 05 05 1
tpr
array0 05 05 1 1
thresholds
array18 08 04 035 01
```

scikitlearn user guide Release 0213

This figure shows an example of such an ROC curve

The `rocaucscore` function computes the area under the receiver operating characteristic ROC curve which is also denoted by AUC or AUROC. By computing the area under the ROC curve the curve information is summarized in one number. For more information see the Wikipedia article on AUC.

```
import numpy as np
from sklearn.metrics import rocaucscore
ytrue = np.array([0, 1, 1])
yscores = np.array([0.1, 0.4, 0.35, 0.8])
rocaucscore(ytrue, yscores)
0.75
```

In multilabel classification the `rocaucscore` function is extended by averaging over the labels as above. Compared to metrics such as the subset accuracy, the Hamming loss, or the F1 score, ROC doesn't require optimizing a threshold for each label. The `rocaucscore` function can also be used in multiclass classification if the predicted outputs have been binarized. In applications where a high false positive rate is not tolerable, the parameter `maxfpr` of `rocaucscore` can be used to summarize the ROC curve up to the given limit.

558 Chapter 3 User Guide

Examples

- See Receiver Operating Characteristic ROC for an example of using ROC to evaluate the quality of the output of a classifier
- See Receiver Operating Characteristic ROC with cross validation for an example of using ROC to evaluate classifier output quality using crossvalidation
- See Species distribution modeling for an example of using ROC to model species distribution

Zero one loss

The `zeroone_loss` function computes the sum or the average of the 01 classification loss $\{0,1\}$ over n samples

By default the function normalizes over the sample To get the sum of the $\{0,1\}$ set `normalize` to `False`

In multilabel classification the `zeroone_loss` scores a subset as one if its labels strictly match the predictions

and as a zero if there are any errors By default the function returns the percentage of imperfectly predicted subsets

To get the count of such subsets instead set `normalize` to `False`

If \hat{y}_i is the predicted value of the i th sample and y_i is the corresponding true value then the 01 loss $\{0,1\}$ is defined as

$$\{0,1\} = 1 - \mathbb{I}(\hat{y}_i = y_i)$$

where \mathbb{I} is the indicator function

```
from sklearn.metrics import zeroone_loss
```

```
ypred = [1, 2, 3, 4]
```

```
ytrue = [2, 2, 3, 4]
```

```
zeroone_loss(ytrue, ypred)
```

0.25

scikitlearn user guide Release 0213
zeroonelossytrue ypred normalize False
1

In the multilabel case with binary label indicators where the first label set 01 has an error
zeroonelossnparray0 1 1 1 npones2 2
05

zeroonelossnparray0 1 1 1 npones2 2 normalize False
1
Example

- See Recursive feature elimination with crossvalidation for an example of zero one loss usage to perform recursive feature elimination with crossvalidation

Brier score loss

Thebrierscoreloss function computes the Brier score for binary classes Quoting Wikipedia

“The Brier score is a proper score function that measures the accuracy of probabilistic predictions It is applicable to tasks in which predictions must assign probabilities to a set of mutually exclusive discrete outcomes”

This function returns a score of the mean square difference between the actual outcome and the predicted probability of the possible outcome The actual outcome has to be 1 or 0 true or false while the predicted probability of the actual outcome can be a value between 0 and 1

The brier score loss is also between 0 to 1 and the lower the score the mean square difference is smaller the more accurate the prediction is It can be thought of as a measure of the “calibration” of a set of probabilistic predictions

$\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$

$\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$

$\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$

where n is the total number of predictions \hat{y}_i is the predicted probability of the actual outcome y_i

Here is a small example of usage of this function

```
import numpy as np
from sklearn.metrics import brierscoreloss

ytrue = np.array([1, 1, 0])
ytruecategorical = np.array(['spam', 'ham', 'ham', 'spam'])
yprob = np.array([0.1, 0.9, 0.8, 0.4])
ypred = np.array([0, 1, 1, 0])
brierscoreloss(ytrue, yprob)
0.055
brierscoreloss(ytrue, 1 - yprob)
0.055
brierscoreloss(ytruecategorical, yprob)
0.055
brierscoreloss(ytrue, yprob * 0.5)
0.0
```

560 Chapter 3 User Guide

Example

- See Probability calibration of classifiers for an example of Brier score loss usage to perform probability calibration of classifiers

References

- G Brier Verification of forecasts expressed in terms of probability Monthly weather review 781 1950

Multilabel ranking metrics

In multilabel learning each sample can have any number of ground truth labels associated with it The goal is to give high scores and better rank to the ground truth labels

Coverage error

Thecoverageerror function computes the average number of labels that have to be included in the final prediction such that all true labels are predicted This is useful if you want to know how many topscoredlabeled you have to predict in average without missing any true one The best value of this metrics is thus the average number of true labels

Note Our implementation’s score is 1 greater than the one given in Tsoumakas et al 2010 This extends it to handle the degenerate case in which an instance has 0 true labels

Formally given a binary indicator matrix of the ground truth labels $y \in \{0,1\}^{n_{\text{samples}} \times n_{\text{labels}}}$ and the score associated with each label $s \in \mathbb{R}^{n_{\text{samples}} \times n_{\text{labels}}}$ the coverage is defined as

$$\text{coverage} = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} \min_{j \in \{1, \dots, n_{\text{labels}}\}} \{s_{ij} \mid y_{ij} = 1\}$$

with rank r_{ij}

$r_{ij} \geq r_{ik}$ Given the rank definition ties in yscores are broken by giving the maximal rank that would have been assigned to all tied values

Here is a small example of usage of this function

```
import numpy as np
from sklearn.metrics import coverageerror

ytrue = np.array([0, 0, 0, 0, 1])
yscore = np.array([0.75, 0.5, 1, 1, 0.2])
coverageerror(ytrue, yscore)
25
```

Label ranking average precision

Thelabelrankingaverageprecisionscore function implements label ranking average precision LRAP This metric is linked to the averageprecisionscore function but is based on the notion of label ranking instead of precision and recall

Label ranking average precision LRAP averages over the samples the answer to the following question for each ground truth label what fraction of higherranked labels were true labels This performance measure will be higher if you are able to give better rank to the labels associated with each sample The obtained score is always strictly greater than 0 and the best value is 1 If there is exactly one relevant label per sample label ranking average precision is equivalent to the mean reciprocal rank

Formally given a binary indicator matrix of the ground truth labels $y \in \{0,1\}^{n_{\text{samples}} \times n_{\text{labels}}}$ and the score associated with each label $s \in \mathbb{R}^{n_{\text{samples}} \times n_{\text{labels}}}$ the average precision is defined as

$$\text{LRAP} = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} \frac{1}{\text{rank}_i} \sum_{j=1}^{\text{rank}_i} y_{ij}$$

where rank_i is the rank of the i -th sample, i.e. the number of elements in the set $\{j \mid y_{ij} = 1\}$ and $\|\cdot\|_0$ is the ℓ_0 “norm” which computes the number of nonzero elements in a vector

Here is a small example of usage of this function

```
import numpy as np
from sklearn.metrics import label_ranking_average_precision_score
y_true = np.array([0, 0, 0, 0, 1])
y_score = np.array([0.75, 0.5, 1, 1, 0.2])
label_ranking_average_precision_score(y_true, y_score)
0.416
```

Ranking loss

The `label_ranking_loss` function computes the ranking loss which averages over the samples the number of label pairs that are incorrectly ordered ie true labels have a lower score than false labels weighted by the inverse of the number of ordered pairs of false and true labels The lowest achievable ranking loss is zero

Formally given a binary indicator matrix of the ground truth labels $y \in \{0,1\}^{n_{\text{samples}} \times n_{\text{labels}}}$ and the score associated with each label $s \in \mathbb{R}^{n_{\text{samples}} \times n_{\text{labels}}}$ the ranking loss is defined as

$$\text{Ranking Loss} = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} \frac{1}{\binom{n_{\text{labels}}}{2}} \sum_{j_1 < j_2} (y_{ij_1} - y_{ij_2}) (s_{ij_1} - s_{ij_2})$$

where $\binom{n}{2}$ computes the cardinality of the set ie the number of elements in the set and $\|\cdot\|_0$ is the ℓ_0 “norm” which computes the number of nonzero elements in a vector

Here is a small example of usage of this function

```
import numpy as np
from sklearn.metrics import label_ranking_loss
y_true = np.array([0, 0, 0, 0, 1])
y_score = np.array([0.75, 0.5, 1, 1, 0.2])
label_ranking_loss(y_true, y_score)
0.75
```

With the following prediction we have perfect and minimal loss

```
y_score = np.array([0, 1, 0.2, 0.1, 0.2, 0.9])
label_ranking_loss(y_true, y_score)
0.0
```


References

- Tsoumakas G Katakis I Vlahavas I 2010 Mining multilabel data In Data mining and knowledge discovery handbook pp 667685 Springer US

Regression metrics

The sklearn.metrics module implements several loss score and utility functions to measure regression performance Some of those have been enhanced to handle the multioutput case meansquarederror meanabsoluteerror explainedvariancescore and r2score

These functions have an multioutput keyword argument which specifies the way the scores or losses for each individual target should be averaged The default is uniformaverage which specifies a uniformly weighted mean over outputs If an ndarray of shape noutputs is passed then its entries are interpreted as weights and an according weighted average is returned If multioutput israwvalues is specified then all unaltered individual scores or losses will be returned in an array of shape noutputs Ther2score and explainedvariancescore accept an additional value varianceweighted for themultioutput parameter This option leads to a weighting of each individual score by the variance of the corresponding target variable This setting quantifies the globally captured unscaled variance If the target variables are of different scale then this score puts more importance on well explaining the higher variance variables multioutputvarianceweighted is the default value for r2score for backward compatibility This will be changed to uniformaverage in the future

Explained variance score

The explainedvariancescore computes the explained variance regression score

If \hat{y} is the estimated target output y the corresponding correct target output and σ^2 is Variance the square of the standard deviation then the explained variance is estimated as follow

$$\text{Explained Variance Score} = 1 - \frac{\text{Residual Sum of Squares}}{\text{Total Sum of Squares}}$$

or

The best possible score is 1.0 lower values are worse

Here is a small example of usage of the explainedvariancescore function

```
from sklearn.metrics import explainedvariancescore
ytrue = [3, 0.5, 2, 7]
ypred = [2.5, 0.0, 2, 8]
explainedvariancescore(ytrue, ypred)
0.957
ytrue = [0.5, 1, 1, 1, 7, 6]
ypred = [0, 2, 1, 2, 8, 5]
explainedvariancescore(ytrue, ypred, multioutput='raw_values')
```

```
array(0.967, 1)
explainedvariancescore(ytrue, ypred, multioutput='raw_values')
```

Max error

The `maxerror` function computes the maximum residual error a metric that captures the worst case error between the predicted value and the true value In a perfectly fitted single output regression model `maxerror` would be 0 on the training set and though this would be highly unlikely in the real world this metric shows the extent of error that the model had when it was fitted

If \hat{y}_i is the predicted value of the i th sample and y_i is the corresponding true value then the max error is defined as

$$\text{Max Error} = \max_i | \hat{y}_i - y_i |$$

Here is a small example of usage of the `maxerror` function

```
from sklearn.metrics import maxerror
ytrue = [3, 2, 7, 1]
ypred = [9, 2, 7, 1]
maxerror(ytrue, ypred)
6
```

The `maxerror` does not support multioutput

Mean absolute error

The `meanabsoluteerror` function computes mean absolute error a risk metric corresponding to the expected value of the absolute error loss or ℓ_1 norm loss

If \hat{y}_i is the predicted value of the i th sample and y_i is the corresponding true value then the mean absolute error

MAE estimated over n samples is defined as

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n | \hat{y}_i - y_i |$$

Here is a small example of usage of the `meanabsoluteerror` function

```
from sklearn.metrics import meanabsoluteerror
ytrue = [3, 0.5, 2, 7]
ypred = [2.5, 0.0, 2, 8]
meanabsoluteerror(ytrue, ypred)
0.5
ytrue = [0.5, 1, 1, 1, 7, 6]
ypred = [0, 2, 1, 2, 8, 5]
meanabsoluteerror(ytrue, ypred)
0.75
meanabsoluteerror(ytrue, ypred, multioutput='raw_values')
array([0.5, 1])
meanabsoluteerror(ytrue, ypred, multioutput=(0.3, 0.7))
```

085

Mean squared error

The `meansquarederror` function computes mean square error a risk metric corresponding to the expected value of the squared quadratic error or loss

If \hat{y}_i is the predicted value of the i th sample and y_i is the corresponding true value then the mean squared error MSE estimated over n samples is defined as

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Here is a small example of usage of the meansquarederror function

```
from sklearn.metrics import meansquarederror
ytrue = 3 05 2 7
ypred = 25 00 2 8
meansquarederror(ytrue, ypred)
0.375
ytrue = 05 1 1 1 7 6
ypred = 0 2 1 2 8 5
meansquarederror(ytrue, ypred)
0.7083
```

- Examples
- See Gradient Boosting regression for an example of mean squared error usage to evaluate gradient boosting regression

Mean squared logarithmic error

The meansquaredlogerror function computes a risk metric corresponding to the expected value of the squared logarithmic quadratic error or loss

If \hat{y}_i is the predicted value of the i th sample and y_i is the corresponding true value then the mean squared logarithmic error MSLE estimated over n samples is defined as

$$MSLE = \frac{1}{n} \sum_{i=1}^n (\log y_i - \log \hat{y}_i)^2$$

Where \log means the natural logarithm of x . This metric is best to use when targets having exponential growth such as population counts average sales of a commodity over a span of years etc. Note that this metric penalizes an underpredicted estimate greater than an overpredicted estimate.

Here is a small example of usage of the meansquaredlogerror function

```
from sklearn.metrics import meansquaredlogerror
ytrue = 3 5 25 7
ypred = 25 5 4 8
meansquaredlogerror(ytrue, ypred)
0.039
ytrue = 05 1 1 2 7 6
ypred = 05 2 1 25 8 8
meansquaredlogerror(ytrue, ypred)
0.044
```

Median absolute error

The medianabsoluteerror is particularly interesting because it is robust to outliers. The loss is calculated by taking the median of all absolute differences between the target and the prediction.

If \hat{y}_i is the predicted value of the i th sample and y_i is the corresponding true value then the median absolute error

MedAE estimated over n samples is defined as

$$\text{MedAE} = \text{median}(|\hat{y}_1 - y_1|, \dots, |\hat{y}_n - y_n|)$$

The medianabsoluteerror does not support multioutput

Here is a small example of usage of the medianabsoluteerror function

```
from sklearn.metrics import medianabsoluteerror
ytrue = 3 05 2 7
ypred = 25 00 2 8
medianabsoluteerror(ytrue, ypred)
05
```

R2score the coefficient of determination

The r2score function computes the coefficient of determination usually denoted as R^2

It represents the proportion of variance of y that has been explained by the independent variables in the model. It provides an indication of goodness of fit and therefore a measure of how well unseen samples are likely to be predicted by the model through the proportion of explained variance.

As such variance is dataset dependent R^2 may not be meaningfully comparable across different datasets. Best possible score is 1.0 and it can be negative because the model can be arbitrarily worse. A constant model that always predicts the expected value of y , disregarding the input features, would get a R^2 score of 0.0.

If \hat{y}_i is the predicted value of the i th sample and y_i is the corresponding true value for total n samples the estimated R^2 is defined as

$$R^2 = 1 - \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

where $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$

Note that r2score calculates unadjusted R^2 without correcting for bias in sample variance of y

Here is a small example of usage of the r2score function

```
from sklearn.metrics import r2score
ytrue = 3 05 2 7
ypred = 25 00 2 8
r2score(ytrue, ypred)
0.948
ytrue = 0.5 1 1 1 7 6
ypred = 0 2 1 2 8 5
r2score(ytrue, ypred, multioutput='variance_weighted')
```

```
0.938
ytrue = 0.5 1 1 1 7 6
ypred = 0 2 1 2 8 5
r2score(ytrue, ypred, multioutput='uniform_average')
```

```
0.936
r2score(ytrue, ypred, multioutput='raw_values')
```

```
array(0.965, 0.908)
r2score(ytrue, ypred, multioutput=(0.3, 0.7))
566 Chapter 3 User Guide
```

0925

Example

- See Lasso and Elastic Net for Sparse Signals for an example of R2score usage to evaluate Lasso and Elastic Net on sparse signals

Clustering metrics

The sklearn.metrics module implements several loss score and utility functions For more information see the Clustering performance evaluation section for instance clustering and Biclustering evaluation for biclustering

Dummy estimators

When doing supervised learning a simple sanity check consists of comparing one's estimator against simple rules of thumb DummyClassifier implements several such simple strategies for classification

- stratified generates random predictions by respecting the training set class distribution
- mostfrequent always predicts the most frequent label in the training set
- prior always predicts the class that maximizes the class prior like mostfrequent and predict_proba returns the class prior
- uniform generates predictions uniformly at random
- constant always predicts a constant label that is provided by the user A major motivation of this method is F1scoring when the positive class is in the minority

Note that with all these strategies the predict method completely ignores the input data

To illustrate DummyClassifier first let's create an imbalanced dataset

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
iris = load_iris
X, y = iris.data, iris.target
y[y == 1] = 0
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
```

Next let's compare the accuracy of SVC (Support Vector Classification) and mostfrequent

```
from sklearn.dummy import DummyClassifier
from sklearn.svm import SVC
clf = SVC(kernel='linear', C=1)
clf.fit(X_train, y_train)
clf.score(X_test, y_test)
```

```
0.63
clf = DummyClassifier(strategy='mostfrequent', random_state=0)
clf.fit(X_train, y_train)
```

```
DummyClassifier(constant=None, random_state=0, strategy='mostfrequent')
clf.score(X_test, y_test)
```

0.57

We see that SVC (Support Vector Classification) doesn't do much better than a dummy classifier Now let's change the kernel

```
scikitlearn user guide Release 0213
clf SVCgamma scale kernelrbf C1fitXtrain ytrain
clf scoreXtest ytest
094
```

We see that the accuracy was boosted to almost 100 A cross validation strategy is recommended for a better estimate of the accuracy if it is not too CPU costly For more information see the Crossvalidation evaluating estimator performance section Moreover if you want to optimize over the parameter space it is highly recommended to use an appropriate methodology see the Tuning the hyperparameters of an estimator section for details More generally when the accuracy of a classifier is too close to random it probably means that something went wrong features are not helpful a hyperparameter is not correctly tuned the classifier is suffering from class imbalance etc DummyRegressor also implements four simple rules of thumb for regression

- mean always predicts the mean of the training targets
 - median always predicts the median of the training targets
 - quantile always predicts a user provided quantile of the training targets
 - constant always predicts a constant value that is provided by the user
- In all these strategies the predict method completely ignores the input data

334 Model persistence

After training a scikitlearn model it is desirable to have a way to persist the model for future use without having to retrain The following section gives you an example of how to persist a model with pickle We'll also review a few security and maintainability issues when working with pickle serialization

An alternative to pickling is to export the model to another format using one of the model export tools listed under Related Projects Unlike pickling once exported you cannot recover the full Scikitlearn estimator object but you can deploy the model for prediction usually by using tools supporting open model interchange formats such as 'ONNX' or 'PMML'

Persistence example

It is possible to save a model in scikitlearn by using Python's builtin persistence model namely pickle

```
from sklearn import svm
from sklearn import datasets
clf svmSVCgamma scale
iris datasetsloadiris
X y irisdata iristarget
clf fitX y
SVCC10 cachesize200 classweightNone coef000
decisionfunctionshapeovr degree3 gamma scale kernelrbf
maxiter1 probabilityFalse randomstateNone shrinkingTrue
tol0001 verboseFalse
import pickle
s pickledumpsclf
clf2 pickleloadss
clf2predictX01
array0
y0
0
```

scikitlearn user guide Release 0213

In the specific case of scikitlearn it may be better to use joblib’s replacement of pickle dump load which is more efficient on objects that carry large numpy arrays internally as is often the case for fitted scikitlearn estimators but can only pickle to the disk and not to a string

```
from joblib import dump load
dumpclf filenamejoblib
```

Later you can load back the pickled model possibly in another Python process with
clf loadfilenamejoblib

Notedump andload functions also accept filelike object instead of filenames More information on data persis
tence with Joblib is available here

Security maintainability limitations

pickle and joblib by extension has some issues regarding maintainability and security Because of this

- Never unpickle untrusted data as it could lead to malicious code being executed upon loading
- While models saved using one version of scikitlearn might load in other versions this is entirely unsupported and inadvisable It should also be kept in mind that operations performed on such data could give different and unexpected results

In order to rebuild a similar model with future versions of scikitlearn additional metadata should be saved along the pickled model

- The training data eg a reference to an immutable snapshot
- The python source code used to generate the model
- The versions of scikitlearn and its dependencies
- The cross validation score obtained on the training data

This should make it possible to check that the crossvalidation score is in the same range as before

Since a model internal representation may be different on two different architectures dumping a model on one archi
tecture and loading it on another architecture is not supported

If you want to know more about these issues and explore other possible serialization methods please refer to this talk
by Alex Gaynor

335 Validation curves plotting scores to evaluate models

Every estimator has its advantages and drawbacks Its generalization error can be decomposed in terms of bias
variance and noise The bias of an estimator is its average error for different training sets The variance of an
estimator indicates how sensitive it is to varying training sets Noise is a property of the data

In the following plot we see a function $y = \cos(3x)$

2D and some noisy samples from that function We use three
different estimators to fit the function linear regression with polynomial features of degree 1 4 and 15 We see that
the first estimator can at best provide only a poor fit to the samples and the true function because it is too simple
high bias the second estimator approximates it almost perfectly and the last estimator approximates the training data
perfectly but does not fit the true function very well ie it is very sensitive to varying training data high variance
Bias and variance are inherent properties of estimators and we usually have to select learning algorithms and hyper
parameters so that both bias and variance are as low as possible see Biasvariance dilemma Another way to reduce

33 Model selection and evaluation 569

the variance of a model is to use more training data However you should only collect more training data if the true function is too complex to be approximated by an estimator with a lower variance
In the simple onedimensional problem that we have seen in the example it is easy to see whether the estimator suffers from bias or variance However in highdimensional spaces models can become very difficult to visualize For this reason it is often helpful to use the tools described below

Examples

- Underfitting vs Overfitting
- Plotting Validation Curves
- Plotting Learning Curves

Validation curve

To validate a model we need a scoring function see Model evaluation quantifying the quality of predictions for example accuracy for classifiers The proper way of choosing multiple hyperparameters of an estimator are of course grid search or similar methods see Tuning the hyperparameters of an estimator that select the hyperparameter with the maximum score on a validation set or multiple validation sets Note that if we optimized the hyperparameters based on a validation score the validation score is biased and not a good estimate of the generalization any longer To get a proper estimate of the generalization we have to compute the score on another test set

However it is sometimes helpful to plot the influence of a single hyperparameter on the training score and the validation score to find out whether the estimator is overfitting or underfitting for some hyperparameter values

The function validationcurve can help in this case

```
import numpy as np
from sklearn.model_selection import validationcurve
from sklearn.datasets import loadiris
from sklearn.linear_model import Ridge
np.random.seed(0)
iris = loadiris()
X, y = iris.data, iris.target
indices = np.arange(y.shape[0])
np.random.shuffle(indices)
570 Chapter 3 User Guide
```


scikitlearn user guide Release 0213

X y Xindices yindices

trainscores validscores validationcurveRidge X y alpha

nplogspace7 3 3

cv5

trainscores

array093 094 092 091 092

093 094 092 091 092

051 052 049 047 049

validscores

array090 084 094 096 093

090 084 094 096 093

046 025 050 049 052

If the training score and the validation score are both low the estimator will be underfitting If the training score is high and the validation score is low the estimator is overfitting and otherwise it is working very well A low training score and a high validation score is usually not possible All three cases can be found in the plot below where we vary the parameter α of an SVM (Support Vector Machine) on the digits dataset

Learning curve

A learning curve shows the validation and training score of an estimator for varying numbers of training samples It is a tool to find out how much we benefit from adding more training data and whether the estimator suffers more from a variance error or a bias error If both the validation score and the training score converge to a value that is too low with increasing size of the training set we will not benefit much from more training data In the following plot you can see an example naive Bayes roughly converges to a low score

We will probably have to use an estimator or a parametrization of the current estimator that can learn more complex concepts ie has a lower bias If the training score is much greater than the validation score for the maximum number of training samples adding more training samples will most likely increase generalization In the following plot you can see that the SVM (Support Vector Machine) could benefit from more training examples

We can use the function learningcurve to generate the values that are required to plot such a learning curve number of samples that have been used the average scores on the training sets and the average scores on the validation sets

from sklearnmodelselection import learningcurve

from sklearnsvm import SVC (Support Vector Classification)

33 Model selection and evaluation 571

scikitlearn user guide Release 0213

trainsizes trainscores validscores learningcurve

SVCkernellinear X y trainsizes50 80 110 cv5

trainsizes

array 50 80 110

trainscores

array098 098 098 098 098

098 1 098 098 098

098 1 098 098 099

validscores

array1 093 1 1 096

1 096 1 1 096

1 096 1 1 096

34 Inspection

341 Partial dependence plots

Partial dependence plots PDP show the dependence between the target response1and a set of ‘target’ features marginalizing over the values of all other features the ‘complement’ features Intuitively we can interpret the partial dependence as the expected target response as a function of the ‘target’ features

Due to the limits of human perception the size of the target feature set must be small usually one or two thus the target features are usually chosen among the most important features

The figure below shows four oneway and one twoway partial dependence plots for the California housing dataset with aGradientBoostingRegressor

Oneway PDPs tell us about the interaction between the target response and the target feature eg linear nonlinear

The upper left plot in the above figure shows the effect of the median income in a district on the median house price

1For classification the target response may be the probability of a class the positive class for binary classification or the decision function

34 Inspection 573

scikitlearn user guide Release 0213

we can clearly see a linear relationship among them Note that PDPs assume that the target features are independent from the complement features and this assumption is often violated in practice

PDPs with two target features show the interactions among the two features For example the twovariable PDP in the above figure shows the dependence of median house price on joint values of house age and average occupants per household We can clearly see an interaction between the two features for an average occupancy greater than two the house price is nearly independent of the house age whereas for values less than 2 there is a strong dependence on age The sklearninspection module provides a convenience function plotpartialdependence to create

oneway and twoway partial dependence plots In the below example we show how to create a grid of partial dependence plots two oneway PDPs for the features 0 and 1 and a twoway PDP between the two features

```
from sklearn.datasets import make_hastie102
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.inspection import plot_partial_dependence
X, y = make_hastie102(random_state=0)
clf = GradientBoostingClassifier(n_estimators=100, learning_rate=10,
                                max_depth=1, random_state=0)
fit = clf.fit(X, y)
features = [0, 1, 0, 1]
plot_partial_dependence(clf, X, features)
```

You can access the newly created figure and Axes objects using plt.gcf() and plt.gca()

For multiclass classification you need to set the class label for which the PDPs should be created via the target argument

```
from sklearn.datasets import load_iris
iris = load_iris()
mc_clf = GradientBoostingClassifier(n_estimators=10,
                                    max_depth=1)
fit = mc_clf.fit(iris.data, iris.target)
features = [3, 2, 3, 2]
plot_partial_dependence(mc_clf, X, features, target=0)
```

The same parameter target is used to specify the target in multioutput regression settings

If you need the raw values of the partial dependence function rather than the plots you can use the sklearn

inspection.partialdependence function

```
from sklearn.inspection import partial_dependence
pdp, axes = partial_dependence(clf, X, [0])
pdp
array(2466, 2466)
axes
array(1624, 1592)
```

The values at which the partial dependence should be evaluated are directly generated from X For 2way partial dependence a 2D grid of values is generated The values field returned by sklearn.inspection.partialdependence gives the actual values used in the grid for each target feature They also correspond to the axis of the plots

For each value of the 'target' features in the grid the partial dependence function needs to marginalize the predictions of the estimator over all possible values of the 'complement' features With the brute method this is done by replacing every target feature value of X by those in the grid and computing the average prediction In decision trees this can be evaluated efficiently without reference to the training data recursion method For each grid point a weighted tree traversal is performed if a split node involves a 'target' feature the corresponding left or right branch is followed otherwise both branches are followed each branch is weighted by the fraction of

scikitlearn user guide Release 0213

training samples that entered that branch Finally the partial dependence is given by a weighted average of all visited leaves Note that with the recursion method X is only used to generate the grid not to compute the averaged predictions The averaged predictions will always be computed on the data with which the trees were trained

Examples

- Partial Dependence Plots

References

35 Dataset transformations

scikitlearn provides a library of transformers which may clean see Preprocessing data reduce see Unsupervised dimensionality reduction expand see Kernel Approximation or generate see Feature extraction feature representations

Like other estimators these are represented by classes with a fit method which learns model parameters eg mean and standard deviation for normalization from a training set and a transform method which applies this transformation model to unseen data fittransform may be more convenient and efficient for modelling and transforming the training data simultaneously

Combining such transformers either in parallel or series is covered in Pipelines and composite estimators Pair wise metrics Affinities and Kernels covers transforming feature spaces into affinity matrices while Transforming the prediction target y considers transformations of the target space eg categorical labels for use in scikitlearn

351 Pipelines and composite estimators

Transformers are usually combined with classifiers regressors or other estimators to build a composite estimator The most common tool is a Pipeline Pipeline is often used in combination with FeatureUnion which concatenates the output of transformers into a composite feature space TransformedTargetRegressor deals with transforming the target ie logtransform y In contrast Pipelines only transform the observed data X

Pipeline chaining estimators

Pipeline can be used to chain multiple estimators into one This is useful as there is often a fixed sequence of steps in processing the data for example feature selection normalization and classification Pipeline serves multiple purposes here

Convenience and encapsulation You only have to call fit and predict once on your data to fit a whole sequence of estimators

Joint parameter selection You can grid search over parameters of all estimators in the pipeline at once

Safety Pipelines help avoid leaking statistics from your test data into the trained model in crossvalidation by ensuring that the same samples are used to train the transformers and predictors

All estimators in a pipeline except the last one must be transformers ie must have a transform method The last estimator may be any type transformer classifier etc

35 Dataset transformations 575

Usage

Construction

ThePipeline is built using a list of key value pairs where the key is a string containing the name you want to give this step and value is an estimator object

```
from sklearnpipeline import Pipeline
from sklearnsvm import SVC (Support Vector Classification)
from sklearndecomposition import PCA (Principal Component Analysis)
estimators reducedim PCA (Principal Component Analysis) clf SVC (Support Vector Classification)
pipe Pipelineestimators
pipe
PipelinememoryNone
stepsreducedim PCAcopyTrue
clf SVCC10 verboseFalse
```

The utility function makepipeline is a shorthand for constructing pipelines it takes a variable number of estimators and returns a pipeline filling in the names automatically

```
from sklearnpipeline import makepipeline
from sklearnnaivebayes import MultinomialNB
from sklearnpreprocessing import Binarizer
makepipelineBinarizer MultinomialNB
PipelinememoryNone
stepsbinarizer BinarizercopyTrue threshold00
multinomialnb MultinomialNBalpha10
classpriorNone
fitpriorTrue
verboseFalse
```

Accessing steps

The estimators of a pipeline are stored as a list in the steps attribute but can be accessed by index or name by indexing with idx the Pipeline

```
pipesteps0
reducedim PCAcopyTrue iteratedpowerauto ncomponentsNone
randomstateNone svdsolverauto tol00
whitenFalse
```

```
pipe0
PCAcopyTrue iteratedpowerauto ncomponentsNone randomstateNone
svdsolverauto tol00 whitenFalse
```

```
pipereducedim
PCAcopyTrue
```

Pipeline’snamedsteps attribute allows accessing steps by name with tab completion in interactive environments
pipenamedstepsreducedim ispipereducedim
True

A subpipeline can also be extracted using the slicing notation commonly used for Python Sequences such as lists or strings although only a step of 1 is permitted This is convenient for performing only some of the transformations or their inverse

scikitlearn user guide Release 0213

```
pipe1
Pipeline(memory=None, steps=[Step1, PCA(copy=True)
```

```
pipe1
Pipeline(memory=None, steps=[Step1, SVC(C10)
```

Nested parameters
Parameters of the estimators in the pipeline can be accessed using the estimatorparameter syntax

```
pipe.set_params(C10)
Pipeline(memory=None,
steps=[Step1, PCA(copy=True, iterated_power=auto)
clf = SVC(C10, cache_size=200, class_weight=None,
verbose=False)
```

This is particularly important for doing grid searches

```
from sklearn.model_selection import GridSearchCV
param_grid = {'Step1__C10': [2, 5, 10],
               'Step1__C01': [10, 100]}
```

```
grid_search = GridSearchCV(pipe, param_grid, param_grid)
```

Individual steps may also be replaced as parameters and nonfinal steps may be ignored by setting them to passthrough

```
from sklearn.linear_model import LogisticRegression
param_grid = {'Step1__C10': [5, 10],
               'Step2__C10': [5, 10],
               'Step3__C10': [5, 10]}
```

```
grid_search = GridSearchCV(pipe, param_grid, param_grid)
```

The estimators of the pipeline can be retrieved by index

```
pipe[0]
pipe.steps[0][1]
pipe.steps[0][1].get_params()
pipe.steps[0][1].set_params()
```

- Examples
- Pipeline Anova SVM (Support Vector Machine)
 - Sample pipeline for text feature extraction and evaluation
 - Pipelining chaining a PCA (Principal Component Analysis) and a logistic regression
 - Explicit feature map approximation for RBF (Radial Basis Function) kernels
 - SVMAnova SVM (Support Vector Machine) with univariate feature selection
 - Selecting dimensionality reduction with Pipeline and GridSearchCV

35 Dataset transformations 577

scikitlearn user guide Release 0213

See also

- Tuning the hyperparameters of an estimator

Notes

Callingfit on the pipeline is the same as calling fit on each estimator in turn transform the input and pass it on to the next step The pipeline has all the methods that the last estimator in the pipeline has ie if the last estimator is a classifier the Pipeline can be used as a classifier If the last estimator is a transformer again so is the pipeline Caching transformers avoid repeated computation

Fitting transformers may be computationally expensive With its memory parameter set Pipeline will cache each transformer after calling fit This feature is used to avoid computing the fit transformers within a pipeline if the parameters and input data are identical A typical example is the case of a grid search in which the transformers can be fitted only once and reused for each configuration

The parameter memory is needed in order to cache the transformers memory can be either a string containing the directory where to cache the transformers or a joblibMemory object

```
from tempfile import mkdtemp
from shutil import rmtree
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.svm import SVC (Support Vector Classification)
from sklearn.pipeline import Pipeline
estimators = reducedim PCA (Principal Component Analysis) clf SVC (Support Vector Classification)
cachedir = mkdtemp
pipe = Pipeline(estimators, memory=cachedir)
pipe
Pipeline
steps=reducedim PCA(copy=True)
clf = SVCC10 verbose=False
Clear the cache directory when you dont need it anymore
rmtree(cachedir)
```

Warning Side effect of caching transformers

Using aPipeline without cache enabled it is possible to inspect the original instance such as

```
from sklearn.datasets import load_digits
digits = load_digits
pca1 = PCA (Principal Component Analysis)
svm1 = SVC(gamma=scale)
pipe = Pipeline(reducedim(pca1), clf=svm1)
pipe.fit(digits.data, digits.target)
```

Pipeline.memory=None

```
steps=reducedim PCA (Principal Component Analysis) clf SVC (Support Vector Classification)
verbose=False
```

The pca instance can be inspected directly

```
print(pca.components_
177484909e19 407058917e18
```


scikitlearn user guide Release 0213

Enabling caching triggers a clone of the transformers before fitting Therefore the transformer instance given to the pipeline cannot be inspected directly In following example accessing the PCA (Principal Component Analysis) instance `pca2` will raise an `AttributeError` since `pca2` will be an unfitted transformer Instead use the attribute named `steps` to inspect estimators within the pipeline

```
cachedir = makedirs
pca2 = PCA (Principal Component Analysis)
svm2 = SVCgamma=scale
cachedpipe = Pipeline(reducedim=pca2, clf=svm2)
memorycachedir =
cachedpipe.fit(digits.data, digits.target)
```

`Pipeline(memory=steps=reducedim.PCA (Principal Component Analysis), clf=SVC (Support Vector Classification), verbose=False, print_cached_pipe_named_steps=reducedim.components)`

177484909e19 407058917e18

Remove the cache directory

`rmtree(cachedir)`

Examples

•Selecting dimensionality reduction with Pipeline and GridSearchCV

Transforming target in regression

`TransformedTargetRegressor` transforms the targets `y` before fitting a regression model The predictions are mapped back to the original space via an inverse transform It takes as an argument the regressor that will be used for prediction and the transformer that will be applied to the target variable

```
import numpy as np
from sklearn.datasets import load_boston
from sklearn.compose import TransformedTargetRegressor
from sklearn.preprocessing import QuantileTransformer
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

boston = load_boston()
X, boston_data, y, boston_target =
transformer = QuantileTransformer(output_distribution='normal')
regressor = LinearRegression()
regr = TransformedTargetRegressor(regressor=regressor,
transformer=transformer)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
regr.fit(X_train, y_train)
TransformedTargetRegressor
print(R2_score(y_test, regr.predict(X_test)))
R2 score 0.67
raw_target_regr = LinearRegression().fit(X_train, y_train)
print(R2_score(y_test, raw_target_regr.predict(X_test)))
R2 score 0.64
```

35 Dataset transformations 579

scikitlearn user guide Release 0213

For simple transformations instead of a Transformer object a pair of functions can be passed defining the transformation and its inverse mapping

```
def funcx
return nplogx
def inversefuncx
return npexpx
```

Subsequently the object is created as

```
regr = TransformedTargetRegressor(regressor=regressor
func=func
inversefunc=inversefunc
regr.fit(Xtrain, ytrain)
TransformedTargetRegressor
print(R2_score(0.2, format(regressor.score(Xtest, ytest)
R2_score(0.65)
```

By default the provided functions are checked at each fit to be the inverse of each other. However it is possible to bypass this checking by setting `check_inverse` to `False`

```
def inversefuncx
return x
regr = TransformedTargetRegressor(regressor=regressor
func=func
inversefunc=inversefunc
check_inverse=False
regr.fit(Xtrain, ytrain)
TransformedTargetRegressor
print(R2_score(0.2, format(regressor.score(Xtest, ytest)
R2_score(0.45)
```

Note: The transformation can be triggered by setting either `transformer` or the pair of functions `func` and `inversefunc`. However setting both options will raise an error.

Examples

- Effect of transforming the targets in regression model

FeatureUnion composite feature spaces

FeatureUnion combines several transformer objects into a new transformer that combines their output. A

FeatureUnion takes a list of transformer objects. During fitting each of these is fit to the data independently.

The transformers are applied in parallel and the feature matrices they output are concatenated side-by-side into a larger matrix.

When you want to apply different transformations to each field of the data see the related class `sklearn.compose`

`ColumnTransformer` see user guide

FeatureUnion serves the same purposes as **Pipeline** for convenience and joint parameter estimation and validation.

FeatureUnion and **Pipeline** can be combined to create complex models.

scikitlearn user guide Release 0213

AFeatureUnion has no way of checking whether two transformers might produce identical features It only produces a union when the feature sets are disjoint and making sure they are the caller’s responsibility Usage

AFeatureUnion is built using a list of key value pairs where the key is the name you want to give to a given transformation an arbitrary string it only serves as an identifier and value is an estimator object

```
from sklearn.pipeline import FeatureUnion
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.decomposition import KernelPCA
estimators = linearpca PCA (Principal Component Analysis) kernelpca KernelPCA
combined = FeatureUnion(estimators
combined
```

```
FeatureUnion(njobs=None
transformerlist=linearpca PCA(copy=True)
kernelpca KernelPCA(alpha=10)
transformerweights=None verbose=False
```

Like pipelines feature unions have a shorthand constructor called makeunion that does not require explicit naming of the components
LikePipeline individual steps may be replaced using setparams and ignored by setting to drop
combinedsetparamskernelpcadrop

```
FeatureUnion(njobs=None
transformerlist=linearpca PCA(copy=True)
kernelpca drop
transformerweights=None verbose=False
```

Examples
•Concatenating multiple feature extraction methods
ColumnTransformer for heterogeneous data

Warning ThecomposeColumnTransformer class is experimental and the API (Application Programming Interface) is subject to change
Many datasets contain features of different types say text floats and dates where each type of feature requires separate preprocessing or feature extraction steps Often it is easiest to preprocess data before applying scikitlearn methods for example using pandas Processing your data before passing it to scikitlearn might be problematic for one of the following reasons

- 1 Incorporating statistics from test data into the preprocessors makes crossvalidation scores unreliable known as data leakage for example in the case of scalers or imputing missing values
 - 2 You may want to include the parameters of the preprocessors in a parameter search
- TheColumnTransformer helps performing different transformations for different columns of the data within a Pipeline that is safe from data leakage and that can be parametrized ColumnTransformer works on arrays sparse matrices and pandas DataFrames
35 Dataset transformations 581

scikitlearn user guide Release 0213

To each column a different transformation can be applied such as preprocessing or a specific feature extraction method

```
import pandas as pd
X = pd.DataFrame(
    city=[London, London, Paris, Sallisaw],
    title=[His Last Bow, How Watson Learned the Trick,
           A Moveable Feast, The Grapes of Wrath],
    expertrating=[5, 3, 4, 5],
    userrating=[4, 5, 4, 3])
```

For this data we might want to encode the city column as a categorical variable using preprocessing OneHotEncoder but apply a featureextractiontextCountVectorizer to the title column. As we might use multiple feature extraction methods on the same column we give each transformer a unique name say citycategory and titlebow. By default the remaining rating columns are ignored.

```
remainderdrop
from sklearn.compose import ColumnTransformer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.preprocessing import OneHotEncoder
columntrans = ColumnTransformer(
    citycategory=OneHotEncoder(dtype=int),
    titlebow=CountVectorizer(),
    remainderdrop=True)
columntrans.fit(X)
ColumnTransformer(n_jobs=None, remainderdrop=True, sparse_threshold=0.3,
transformer_weights=None, transformers=[
columntrans.get_feature_names()
```

```
citycategoryx0=London, citycategoryx0=Paris, citycategoryx0=Sallisaw,
titlebowbow, titlebowfeast, titlebowgrapes, titlebowhis,
titlebowhow, titlebowlast, titlebowlearned, titlebowmoveable,
titlebowof, titlebowthe, titlebowtrick, titlebowwatson,
titlebowwrath,
columntranstransformXtoarray
```

```
array([[1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0],
       [1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0],
       [0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
       [0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1]])
```

In the above example the CountVectorizer expects a 1D array as input and therefore the columns were specified as a string 'title'. However preprocessingOneHotEncoder as most of other transformers expects 2D data therefore in that case you need to specify the column as a list of strings ['city']. Apart from a scalar or a single item list the column selection can be specified as a list of multiple items, an integer array, a slice or a boolean mask. Strings can reference columns if the input is a DataFrame, integers are always interpreted as the positional columns.

We can keep the remaining rating columns by setting remainder=pass through. The values are appended to the end of the transformation.

```
columntrans = ColumnTransformer(
    citycategory=OneHotEncoder(dtype=int),
```

```
scikitlearn user guide Release 0213
titlebow CountVectorizer title
remainderpassthrough
columntransfittransformX
```

```
array1 0 0 1 0 0 1 0 1 0 0 0 0 0 0 5 4
1 0 0 0 0 0 0 1 0 1 0 0 1 1 1 0 3 5
0 1 0 0 1 0 0 0 0 0 1 0 0 0 0 0 4 4
0 0 1 0 0 1 0 0 0 0 0 1 1 0 0 1 5 3
```

```
Theremainder parameter can be set to an estimator to transform the remaining rating columns The transformed
values are appended to the end of the transformation
from sklearnpreprocessing import MinMaxScaler
columntrans ColumnTransformer
citycategory OneHotEncoder city
titlebow CountVectorizer title
remainderMinMaxScaler
columntransfittransformX 2
```

```
array1 05
0 1
05 05
1 0
```

```
Themakecolumntransformer function is available to more easily create a ColumnTransformer object
Specifically the names will be given automatically The equivalent for the above example would be
from sklearncompose import makecolumntransformer
columntrans makecolumntransformer
```

```
OneHotEncoder city
CountVectorizer title
remainderMinMaxScaler
columntrans
ColumnTransformernjobsNone remainderMinMaxScalercopyTrue
sparseshreshold03
transformerweightsNone
transformersonehotencoder
```

- Examples
- Column Transformer with Heterogeneous Data Sources
 - Column Transformer with Mixed Types

352 Feature extraction

The sklearnfeatureextraction module can be used to extract features in a format supported by machine learning algorithms from datasets consisting of formats such as text and image

Note Feature extraction is very different from Feature selection the former consists in transforming arbitrary data such as text or images into numerical features usable for machine learning The latter is a machine learning technique

35 Dataset transformations 583

scikitlearn user guide Release 0213
applied on these features
Loading features from dicts
The classDictVectorizer can be used to convert feature arrays represented as lists of standard Python dict objects to the NumPySciPy representation used by scikitlearn estimators
While not particularly fast to process Python’s dict has the advantages of being convenient to use being sparse
absent features need not be stored and storing feature names in addition to values
DictVectorizer implements what is called oneofK or “onehot” coding for categorical aka nominal discrete features Categorical features are “attributevalue” pairs where the value is restricted to a list of discrete of possibilities without ordering eg topic identifiers types of objects tags names
In the following “city” is a categorical attribute while “temperature” is a traditional numerical feature measurements
city Dubai temperature 33
city London temperature 12
city San Francisco temperature 18

```
from sklearn.feature_extraction import DictVectorizer
vec = DictVectorizer
vec.fit_transform(measurements_to_array)
array 1 0 0 33
0 1 0 12
0 0 1 18
vec.get_feature_names
cityDubai cityLondon citySan Francisco temperature
```

DictVectorizer is also a useful representation transformation for training sequence classifiers in Natural Language Processing models that typically work by extracting feature windows around a particular word of interest
For example suppose that we have a first algorithm that extracts Part of Speech PoS tags that we want to use as complementary tags for training a sequence classifier eg a chunker The following dict could be such a window of features extracted around the word ‘sat’ in the sentence ‘The cat sat on the mat’

```
poswindow

word2 the
pos2 DT
word1 cat
pos1 NN
word1 on
pos1 PP
```

in a real application one would extract many such dictionaries

This description can be vectorized into a sparse twodimensional matrix suitable for feeding into a classifier maybe after being piped into a textTfidfTransformer for normalization

```
vec = DictVectorizer
posvectorized = vec.fit_transform(poswindow)
```

584 Chapter 3 User Guide

scikitlearn user guide Release 0213

```
posvectorized
1x6 sparse matrix of type numpyfloat64
with 6 stored elements in Compressed Sparse format
posvectorizedtoarray
array1 1 1 1 1 1
vecgetfeaturenames
pos1PP pos1NN pos2DT word1on word1cat word2the
```

As you can imagine if one extracts such a context around each individual word of a corpus of documents the resulting matrix will be very wide many onehotfeatures with most of them being valued to zero most of the time So as to make the resulting data structure able to fit in memory the DictVectorizer class uses a scipysparse matrix by default instead of a numpyndarray

Feature hashing

The classFeatureHasher is a highspeed lowmemory vectorizer that uses a technique known as feature hashing or the “hashing trick” Instead of building a hash table of the features encountered in training as the vectorizers do instances of FeatureHasher apply a hash function to the features to determine their column index in sample matrices directly The result is increased speed and reduced memory usage at the expense of inspectability the hasher does not remember what the input features looked like and has no inversetransform method

Since the hash function might cause collisions between unrelated features a signed hash function is used and the sign of the hash value determines the sign of the value stored in the output matrix for a feature This way collisions are likely to cancel out rather than accumulate error and the expected mean of any output feature’s value is zero

This mechanism is enabled by default with alternatesignTrue and is particularly useful for small hash table sizes nfeatures 10000 For large hash table sizes it can be disabled to allow the output to be passed to estimators like sklearnnaivebayesMultinomialNB orsklearnfeatureselectionchi2

feature selectors that expect nonnegative inputs

FeatureHasher accepts either mappings like Python’s dict and its variants in the collections module

feature value pairs or strings depending on the constructor parameter inputtype Mapping are treated as lists offeature value pairs while single strings have an implicit value of 1 so feat1 feat2 feat3 is interpreted as feat1 1 feat2 1 feat3 1 If a single feature occurs multiple times in a sample the associated values will be summed so feat 2 andfeat 35 become feat 55 The output from FeatureHasher is always a scipysparse matrix in the CSR (Compressed Sparse Row) format

Feature hashing can be employed in document classification but unlike textCountVectorizer

FeatureHasher does not do word splitting or any other preprocessing except UnicodetoUTF8 encoding see

Vectorizing a large text corpus with the hashing trick below for a combined tokenizerhasher

As an example consider a wordlevel natural language processing task that needs features extracted from token partofspeech pairs One could use a Python generator function to extract features

```
deftokenfeaturestoken partofspeech
iftokenisdigit
yieldnumeric
else
yieldtokenformattokenlower
yieldtokenpos formattoken partofspeech
iftoken0isupper
yielduppercaseinitial
iftokenisupper
yieldalluppercase
yieldposformatpartofspeech
```

Then therawX to be fed to FeatureHashertransform can be constructed using

35 Dataset transformations 585

scikitlearn user guide Release 0213

rawX tokenfeaturestok postaggertok fortokincorpus

and fed to a hasher with

hasher FeatureHasherinputtypestring

X hashertransformrawX

to get asciipysparse matrixX

Note the use of a generator comprehension which introduces laziness into the feature extraction tokens are only

processed on demand from the hasher

Implementation details

FeatureHasher uses the signed 32bit variant of MurmurHash3 As a result and because of limitations in scipy

sparse the maximum number of features supported is currently 231–1

The original formulation of the hashing trick by Weinberger et al used two separate hash functions *hand* to deter

mine the column index and sign of a feature respectively The present implementation works under the assumption

that the sign bit of MurmurHash3 is independent of its other bits

Since a simple modulo is used to transform the hash function to a column index it is advisable to use a power of two

as thenfeatures parameter otherwise the features will not be mapped evenly to the columns

References

• Kilian Weinberger Anirban Dasgupta John Langford Alex Smola and Josh Attenberg 2009 Feature hash

ing for large scale multitask learning Proc ICML

• MurmurHash3

Text feature extraction

The Bag of Words representation

Text Analysis is a major application field for machine learning algorithms However the raw data a sequence of

symbols cannot be fed directly to the algorithms themselves as most of them expect numerical feature vectors with a

fixed size rather than the raw text documents with variable length

In order to address this scikitlearn provides utilities for the most common ways to extract numerical features from

text content namely

•tokenizing strings and giving an integer id for each possible token for instance by using whitespaces and

punctuation as token separators

•counting the occurrences of tokens in each document

•normalizing and weighting with diminishing importance tokens that occur in the majority of samples documents

In this scheme features and samples are defined as follows

• each individual token occurrence frequency normalized or not is treated as a feature

• the vector of all the token frequencies for a given document is considered a multivariate sample

586 Chapter 3 User Guide

scikitlearn user guide Release 0213

A corpus of documents can thus be represented by a matrix with one row per document and one column per token eg word occurring in the corpus

We call vectorization the general process of turning a collection of text documents into numerical feature vectors This specific strategy tokenization counting and normalization is called the Bag of Words or “Bag of ngrams” representation Documents are described by word occurrences while completely ignoring the relative position information of the words in the document

Sparsity

As most documents will typically use a very small subset of the words used in the corpus the resulting matrix will have many feature values that are zeros typically more than 99 of them

For instance a collection of 10000 short text documents such as emails will use a vocabulary with a size in the order of 100000 unique words in total while each document will use 100 to 1000 unique words individually

In order to be able to store such a matrix in memory but also to speed up algebraic operations matrix vector implementations will typically use a sparse representation such as the implementations available in the scipysparse package

Common Vectorizer usage

CountVectorizer implements both tokenization and occurrence counting in a single class

from sklearn.feature_extraction.text import CountVectorizer

This model has many parameters however the default values are quite reasonable please see the reference documentation for the details

vectorizer = CountVectorizer

vectorizer

CountVectorizer(analyzer='word', binary=False, decode_error='strict',

dtype=np.int64, encoding='utf-8', input_content

lowercase=True, max_df=10, max_features=None, min_df=1,

ngram_range=(1, 1), preprocessor=None, stopwords=None,

strip_accents=None, token_pattern=r'\b\w+\b',

tokenizer=None, vocabulary=None)

Let’s use it to tokenize and count the word occurrences of a minimalistic corpus of text documents

corpus

This is the first document

This is the second second document

And the third one

Is this the first document

X = vectorizer.fit_transform(corpus)

X

4x9 sparse matrix of type np.int64

with 19 stored elements in Compressed Sparse format

The default configuration tokenizes the string by extracting words of at least 2 letters The specific function that does this step can be requested explicitly

35 Dataset transformations 587

scikitlearn user guide Release 0213

```
analyze vectorizerbuildanalyzer
analyzeThis is a text document to analyze
this is text document to analyze
```

True
Each term found by the analyzer during the fit is assigned a unique integer index corresponding to a column in the resulting matrix This interpretation of the columns can be retrieved as follows

```
vectorizergetfeaturenames
and document first is one
second the third this
```

```
True
Xtoarray
array0 1 1 1 0 0 1 0 1
0 1 0 1 0 2 1 0 1
1 0 0 0 1 0 1 1 0
0 1 1 1 0 0 1 0 1
```

The converse mapping from feature name to column index is stored in the vocabulary attribute of the vectorizer

```
vectorizervocabularygetdocument
1
Hence words that were not seen in the training corpus will be completely ignored in future calls to the transform
method
vectorizertransformSomething completely newtoarray
```

```
array0 0 0 0 0 0 0 0 0
Note that in the previous corpus the first and the last documents have exactly the same words hence are encoded in
equal vectors In particular we lose the information that the last document is an interrogative form To preserve some
of the local ordering information we can extract 2grams of words in addition to the 1grams individual words
```

```
bigramvectorizer CountVectorizerngramrange1 2
tokenpatternrbwb mindf1
```

```
analyze bigramvectorizerbuildanalyzer
analyzeBigrams are cool
bi grams are cool bi grams grams are are cool
```

True
The vocabulary extracted by this vectorizer is hence much bigger and can now resolve ambiguities encoded in local positioning patterns

```
X2 bigramvectorizerfittransformcorpustoarray
X2
```

```
array0 0 1 1 1 1 1 0 0 0 0 0 1 1 0 0 0 0 1 1 0
0 0 1 0 0 1 1 0 0 2 1 1 1 0 1 0 0 0 1 1 0
1 1 0 0 0 0 0 0 1 0 0 0 1 0 0 1 1 1 0 0 0
0 0 1 1 1 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 1
```

In particular the interrogative form “Is this” is only present in the last document

```
featureindex bigramvectorizervocabularygetis this
X2 featureindex
```

array0 0 0 1

Using stop words

Stop words are words like “and” “the” “him” which are presumed to be uninformative in representing the content of a text and which may be removed to avoid them being construed as signal for prediction Sometimes however similar words are useful for prediction such as in classifying writing style or personality

There are several known issues in our provided ‘english’ stop word list See NQY18

Please take care in choosing a stop word list Popular stop word lists may include words that are highly informative to some tasks such as computer

You should also make sure that the stop word list has had the same preprocessing and tokenization applied as the one used in the vectorizer The word we’ve is split into weandveby CountVectorizer’s default tokenizer so if we’ve is in stopwords but veis not vewill be retained from we’ve in transformed text Our vectorizers will try to identify and warn about some kinds of inconsistencies

References

Tf-idf term weighting

In a large text corpus some words will be very present eg “the” “a” “is” in English hence carrying very little meaningful information about the actual contents of the document If we were to feed the direct count data directly to a classifier those very frequent terms would shadow the frequencies of rarer yet more interesting terms In order to reweight the count features into floating point values suitable for usage by a classifier it is very common to use the tf-idf transform

Tf means termfrequency while tf-idf means termfrequency times inverse documentfrequency $tfd = tf \times idf$

$tfd \times idf$

Using the TfidfTransformer’s default settings TfidfTransformernorml2 useidfTrue

smoothidfTrue sublineartfFalse the term frequency the number of times a term occurs in a given document is multiplied with idf component which is computed as

$idf = \log \frac{1}{p}$

$idf = 1$

where p is the total number of documents in the document set and d is the number of documents in the document set that contain term t The resulting tfidf vectors are then normalized by the Euclidean norm

$\frac{1}{\sqrt{\sum_i tfidf_i^2}}$

$\frac{1}{\sqrt{2}}$

$\frac{1}{\sqrt{2}}$

This was originally a term weighting scheme developed for information retrieval as a ranking function for search engines results that has also found good use in document classification and clustering

The following sections contain further explanations and examples that illustrate how the tfidfs are computed exactly and how the tfidfs computed in scikitlearn’s TfidfTransformer andTfidfVectorizer differ slightly from the standard textbook notation that defines the idf as

$idf = \log \frac{1}{p}$

$idf = 1$

In theTfidfTransformer andTfidfVectorizer withsmoothidfFalse the “1” count is added to the idf instead of the idf’s denominator

$idf = \log \frac{1}{p+1}$

$idf = 1$

scikitlearn user guide Release 0213

This normalization is implemented by the TfidfTransformer class

```
from sklearn.feature_extraction.text import TfidfTransformer
```

```
transformer = TfidfTransformer(smooth_idf=False)
```

```
transformer
```

```
TfidfTransformer(norml2=True, smooth_idf=False, sublinear_tf=False)
```

```
use_idf=True
```

Again please see the reference documentation for the details on all the parameters

Let's take an example with the following counts The first term is present 100 of the time hence not very interesting

The two other features only in less than 50 of the time hence probably more representative of the content of the documents

```
counts = [[3, 0, 1]
```

```
          [2, 0, 0]
```

```
          [3, 0, 0]
```

```
          [4, 0, 0]
```

```
          [3, 2, 0]
```

```
          [3, 0, 2]]
```

```
tfidf = transformer.fit_transform(counts)
```

```
tfidf
```

```
6x3 sparse matrix of type numpy.float64
```

```
with 9 stored elements in Compressed Sparse format
```

```
tfidf.toarray()
```

```
array([[0.81940995, 0.057320793,
```

```
       [1.0, 0.0,
```

```
       [1.0, 0.0,
```

```
       [1.0, 0.0,
```

```
       [0.47330339, 0.088089948, 0.
```

```
       [0.58149261, 0.081355169,
```

```
       [0.58149261, 0.081355169,
```

```
       [0.58149261, 0.081355169,
```

```
       [0.58149261, 0.081355169,
```

For example we can compute the tfidf of the first term in the first document in the counts array as follows

```
df[0,0]
```

```
df[0,0] = 3
```

```
idf[0,0] = log
```

```
df[0,0] = 1 + log(3) = 1.1
```

```
tfidf[0,0] = df[0,0] * idf[0,0] = 3 * 1.1 = 3.3
```

Now if we repeat this computation for the remaining 2 terms in the document we get

```
tfidf[0,1] = 0 * log(6) = 0
```

```
tfidf[0,2] = 1 * log(2) = 1 * 0.693 = 0.693
```

and the vector of raw tfidfs

```
tfidf[0,:] = [3.3, 0, 0.693]
```

Then applying the Euclidean L2 norm we obtain the following tfidfs for document 1

```
3.3^2 + 0^2 + 0.693^2
```

```
3.3^2 + 0^2 + 0.693^2 = 11.1801
```

```
sqrt(11.1801) = 3.3436
```

590 Chapter 3 User Guide

scikitlearn user guide Release 0213

Furthermore the default parameter smoothidfTrue adds “1” to the numerator and denominator as if an extra document was seen containing every term in the collection exactly once which prevents zero divisions

idf= log1=

1df= 1

Using this modification the tfidf of the third term in document 1 changes to 18473

tfidf term3 1×log73 1≈18473

And the L2normalized tfidf changes to

3018473√

3202184732 08515005243

transformer TfIdfTransformer

transformerfittransformcountstoarray

array085151335 0 052433293

1 0 0

1 0 0

1 0 0

055422893 083236428 0

063035731 0 077630514

The weights of each feature computed by the fit method call are stored in a model attribute

transformeridf

array1 225 184

As tf-idf is very often used for text features there is also another class called TfIdfVectorizer that combines all

the options of CountVectorizer andTfidfTransformer in a single model

from sklearnfeatureextractiontext import TfIdfVectorizer

vectorizer TfIdfVectorizer

vectorizerfittransformcorpus

4x9 sparse matrix of type numpyfloat64

with 19 stored elements in Compressed Sparse format

While the tf-idf normalization is often very useful there might be cases where the binary occurrence markers might

offer better features This can be achieved by using the binary parameter of CountVectorizer In particular

some estimators such as Bernoulli Naive Bayes explicitly model discrete boolean random variables Also very short

texts are likely to have noisy tf-idf values while the binary occurrence info is more stable

As usual the best way to adjust the feature extraction parameters is to use a crossvalidated grid search for instance by

pipelining the feature extractor with a classifier

•Sample pipeline for text feature extraction and evaluation

Decoding text files

Text is made of characters but files are made of bytes These bytes represent characters according to some encoding

To work with text files in Python their bytes must be decoded to a character set called Unicode Common encodings

are ASCII Latin1 Western Europe KOI8R Russian and the universal encodings UTF8 and UTF16 Many

others exist

Note An encoding can also be called a ‘character set’ but this term is less accurate several encodings can exist for

a single character set

35 Dataset transformations 591

scikitlearn user guide Release 0213

The text feature extractors in scikitlearn know how to decode text files but only if you tell them what encoding the files are in The CountVectorizer takes anencoding parameter for this purpose For modern text files the correct encoding is probably UTF8 which is therefore the default encodingutf8

If the text you are loading is not actually encoded with UTF8 however you will get a UnicodeDecodeError

The vectorizers can be told to be silent about decoding errors by setting the decodeerror parameter to either ignore orreplace See the documentation for the Python function bytesdecode for more details type helpbytesdecode at the Python prompt

If you are having trouble decoding text here are some things to try

- Find out what the actual encoding of the text is The file might come with a header or README that tells you the encoding or there might be some standard encoding you can assume based on where the text comes from

- You may be able to find out what kind of encoding it is in general using the UNIX command file The Python chardet module comes with a script called chardetectpy that will guess the specific encoding though you cannot rely on its guess being correct

- You could try UTF8 and disregard the errors You can decode byte strings with bytes decodeerrorsreplace to replace all decoding errors with a meaningless character or set decodeerrorreplace in the vectorizer This may damage the usefulness of your features

- Real text may come from a variety of sources that may have used different encodings or even be sloppily decoded in a different encoding than the one it was encoded with This is common in text retrieved from the Web The Python package ftfy can automatically sort out some classes of decoding errors so you could try decoding the unknown text as latin1 and then using ftfy to fix errors

- If the text is in a mishmash of encodings that is simply too hard to sort out which is the case for the 20 Newsgroups dataset you can fall back on a simple singlebyte encoding such as latin1 Some text may display incorrectly but at least the same sequence of bytes will always represent the same feature

For example the following snippet uses chardet not shipped with scikitlearn must be installed separately to figure out the encoding of three texts It then vectorizes the texts and prints the learned vocabulary The output is not shown here

```
import chardet
text1 bSei mir gegr xc3xbxc3x9f t mein Sauerkraut
text2 bholdselig sind deine Ger xfcche
text3 b xffxfe Ax00ux00fx00 x00 Fx00lx00xfcx00 gx00ex00lx00nx00
↵x00dx00ex00sx00 x00 Gx00ex00sx00ax00nx00gx00ex00sx00x00
↵x00Hx00ex00rx00zx00lx00ix00ex00bx00cx00hx00ex00nx00x00
↵x00tx00rx00ax00gx00 x00 ix00cx00hx00 x00 dx00ix00cx00hx00
↵x00fx00ox00rx00tx00
decoded xdecodechardetdetectxencoding
for xintext1 text2 text3
v CountVectorizerfitdecodedvocabulary
for terminv printv
```

Depending on the version of chardet it might get the first one wrong

For an introduction to Unicode and character encodings in general see Joel Spolsky’s Absolute Minimum Every Software Developer Must Know About Unicode

Applications and examples

The bag of words representation is quite simplistic but surprisingly useful in practice

In particular in a supervised setting it can be successfully combined with fast and scalable linear models to train document classifiers for instance

scikitlearn user guide Release 0213

•Classification of text documents using sparse features

In an unsupervised setting it can be used to group similar documents together by applying clustering algorithms such asKmeans

•Clustering text documents using kmeans

Finally it is possible to discover the main topics of a corpus by relaxing the hard assignment constraint of clustering for instance by using Nonnegative matrix factorization NMF (Non-Negative Matrix Factorization) or NNMF

•Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation

Limitations of the Bag of Words representation

A collection of unigrams what bag of words is cannot capture phrases and multiword expressions effectively disregarding any word order dependence Additionally the bag of words model doesn't account for potential misspellings or word derivations

Ngrams to the rescue Instead of building a simple collection of unigrams n1 one might prefer a collection of bigrams n2 where occurrences of pairs of consecutive words are counted

One might alternatively consider a collection of character ngrams a representation resilient against misspellings and derivations

For example let's say we're dealing with a corpus of two documents words wprds The second document contains a misspelling of the word 'words' A simple bag of words representation would consider these two as very distinct documents differing in both of the two possible features A character 2gram representation however would find the documents matching in 4 out of 8 features which may help the preferred classifier decide better ngramvectorizer CountVectorizeranalyzercharwb ngramrange2 2

counts ngramvectorizerfittransformwords wprds

ngramvectorizergetfeaturenames

w ds or pr rd s wo wp

True

countstoarrayastypeint

array1 1 1 0 1 1 1 0

1 1 0 1 1 1 0 1

In the above example charwb analyzer is used which creates ngrams only from characters inside word boundaries padded with space on each side The char analyzer alternatively creates ngrams that span across words

ngramvectorizer CountVectorizeranalyzercharwb ngramrange5 5

ngramvectorizerfittransformjumpy fox

1x4 sparse matrix of type numpyint64

with 4 stored elements in Compressed Sparse format

ngramvectorizergetfeaturenames

fox jump jumpy umpy

True

ngramvectorizer CountVectorizeranalyzerchar ngramrange5 5

ngramvectorizerfittransformjumpy fox

1x5 sparse matrix of type numpyint64

with 5 stored elements in Compressed Sparse format

ngramvectorizergetfeaturenames

jumpy mpy f py fo umpy y fox

True

35 Dataset transformations 593

scikitlearn user guide Release 0213

The word boundariesaware variant charwb is especially interesting for languages that use whitespaces for word separation as it generates significantly less noisy features than the raw char variant in that case For such languages it can increase both the predictive accuracy and convergence speed of classifiers trained using such features while retaining the robustness with regards to misspellings and word derivations While some local positioning information can be preserved by extracting ngrams instead of individual words bag of words and bag of ngrams destroy most of the inner structure of the document and hence most of the meaning carried by that internal structure

In order to address the wider task of Natural Language Understanding the local structure of sentences and paragraphs should thus be taken into account Many such models will thus be casted as “Structured output” problems which are currently outside of the scope of scikitlearn

Vectorizing a large text corpus with the hashing trick

The above vectorization scheme is simple but the fact that it holds an in memory mapping from the string tokens to the integer feature indices thevocabulary attribute causes several problems when dealing with large datasets

- the larger the corpus the larger the vocabulary will grow and hence the memory use too
- fitting requires the allocation of intermediate data structures of size proportional to that of the original dataset
- building the wordmapping requires a full pass over the dataset hence it is not possible to fit text classifiers in a strictly online manner
- pickling and unpickling vectorizers with a large vocabulary can be very slow typically much slower than pickling unpickling flat data structures such as a NumPy array of the same size
- it is not easily possible to split the vectorization work into concurrent sub tasks as the vocabulary attribute would have to be a shared state with a fine grained synchronization barrier the mapping from token string to feature index is dependent on ordering of the first occurrence of each token hence would have to be shared potentially harming the concurrent workers’ performance to the point of making them slower than the sequential variant

It is possible to overcome those limitations by combining the “hashing trick” Feature hashing implemented by the sklearnfeatureextractionFeatureHasher class and the text preprocessing and tokenization features of theCountVectorizer

This combination is implementing in HashingVectorizer a transformer class that is mostly API (Application Programming Interface) compatible with CountVectorizer HashingVectorizer is stateless meaning that you don’t have to call fit on it

from sklearnfeatureextractiontext import HashingVectorizer

hv = HashingVectorizer(nfeatures=10

hvtransformcorpus

4x10 sparse matrix of type numpyfloat64

with 16 stored elements in Compressed Sparse format

You can see that 16 nonzero feature tokens were extracted in the vector output this is less than the 19 nonzeros extracted previously by the CountVectorizer on the same toy corpus The discrepancy comes from hash function collisions because of the low value of the nfeatures parameter

In a real world setting the nfeatures parameter can be left to its default value of 220roughly one million possible features If memory or downstream models size is an issue selecting a lower value such as 218might help without introducing too many additional collisions on typical text classification tasks

scikitlearn user guide Release 0213

Note that the dimensionality does not affect the CPU training time of algorithms which operate on CSR (Compressed Sparse Row) matrices LinearSVCdualTrue Perceptron SGDClassifier PassiveAggressive but it does for algorithms that work with CSC matrices LinearSVCdualFalse Lasso etc

Let's try again with the default setting

```
hv = HashingVectorizer  
hv.transform(corpus
```

4x1048576 sparse matrix of type numpyfloat64
with 19 stored elements in Compressed Sparse format

We no longer get the collisions but this comes at the expense of a much larger dimensionality of the output space Of course other terms than the 19 used here might still collide with each other

TheHashingVectorizer also comes with the following limitations

- it is not possible to invert the model no inversetransform method nor to access the original string representation of the features because of the oneway nature of the hash function that performs the mapping
- it does not provide IDF weighting as that would introduce statefulness in the model A TfidfTransformer can be appended to it in a pipeline if required

Performing outofcore scaling with HashingVectorizer

An interesting development of using a HashingVectorizer is the ability to perform outofcore scaling This means that we can learn from data that does not fit into the computer's main memory

A strategy to implement outofcore scaling is to stream data to the estimator in minibatches Each minibatch is vectorized using HashingVectorizer so as to guarantee that the input space of the estimator has always the same dimensionality The amount of memory used at any time is thus bounded by the size of a minibatch Although there is no limit to the amount of data that can be ingested using such an approach from a practical point of view the learning time is often limited by the CPU time one wants to spend on the task

For a fullfledged example of outofcore scaling in a text classification task see Outofcore classification of text documents

Customizing the vectorizer classes

It is possible to customize the behavior by passing a callable to the vectorizer constructor

```
def mytokenizers  
    return ssplit
```

```
vectorizer = CountVectorizertokenizermymtokenizer  
vectorizer.buildanalyzer(some_punctuation  
    some_punctuation  
    True
```

In particular we name

- preprocessor a callable that takes an entire document as input as a single string and returns a possibly transformed version of the document still as an entire string This can be used to remove HTML tags lowercase the entire document etc
- tokenizer a callable that takes the output from the preprocessor and splits it into tokens then returns a list of these

35 Dataset transformations 595

scikitlearn user guide Release 0213

•analyzer a callable that replaces the preprocessor and tokenizer The default analyzers all call the preprocessor and tokenizer but custom analyzers will skip this Ngram extraction and stop word filtering take place at the analyzer level so a custom analyzer may have to reproduce these steps
Lucene users might recognize these names but be aware that scikitlearn concepts may not map onetooone onto Lucene concepts

To make the preprocessor tokenizer and analyzers aware of the model parameters it is possible to derive from the class and override the buildpreprocessor buildtokenizer andbuildanalyzer factory methods instead of passing custom functions

Some tips and tricks

• If documents are pretokenized by an external package then store them in files or strings with the tokens separated by whitespace and pass analyzerstrsplit
• Fancy tokenlevel analysis such as stemming lemmatizing compound splitting filtering based on partof speech etc are not included in the scikitlearn codebase but can be added by customizing either the tokenizer or the analyzer Here’s a CountVectorizer with a tokenizer and lemmatizer using NLTK

```
from nltk import wordtokenize
from nltkstem import WordNetLemmatizer
class LemmaTokenizer object
def initself
selfwnl WordNetLemmatizer
def callself doc
return selfwnllemmatizet fortinwordtokenizedoc
```

vect CountVectorizertokenizerLemmaTokenizer

Note that this will not filter out punctuation

The following example will for instance transform some British spelling to American spelling

```
import re
def tobritishtokens
for tintokens
t resubrou r1or t
t resubrbtre r1er t
t resubriyseingation r1z2 t
t resubrogue og t
yield t
```

```
class CustomVectorizer CountVectorizer
def buildtokenizerself
tokenize superbuilttokenizer
return lambda doc listtobritishtokenizedoc
```

```
printCustomVectorizerbuildanalyzerucolor colour
color color
```

for other styles of preprocessing examples include stemming lemmatization or normalizing numerical tokens with the latter illustrated in

-Biclustering documents with the Spectral Coclustering algorithm

Customizing the vectorizer can also be useful when handling Asian languages that do not use an explicit word separator such as whitespace

scikitlearn user guide Release 0213

Image feature extraction

Patch extraction

Theextractpatches2d function extracts patches from an image stored as a twodimensional array or threedimensional with color information along the third axis For rebuilding an image from all its patches use reconstructfrompatches2d For example let use generate a 4x4 pixel picture with 3 color channels eg in RGB format

```
import numpy as np
from sklearnfeatureextraction import image
oneimage = np.arange(4 * 4 * 3).reshape(4, 4, 3)
oneimage[0, 0, 0] = 12 # R channel of a fake RGB picture
oneimage[0, 0, 1] = 15
oneimage[0, 0, 2] = 18
oneimage[0, 1, 0] = 24
oneimage[0, 1, 1] = 27
oneimage[0, 1, 2] = 30
oneimage[0, 2, 0] = 36
oneimage[0, 2, 1] = 39
oneimage[0, 2, 2] = 42
oneimage[0, 3, 0] = 45
patches = imageextractpatches2d(oneimage, 2, 2, max_patches=20,
                                random_state=0)
patches.shape
(2, 2, 2, 3)
patches[0, 0, 0, 0]
array([0, 3])
patches[0, 0, 0, 1]
array([12, 15])
patches[0, 0, 0, 2]
array([18, 27])
patches[0, 0, 1, 0]
array([15, 24])
patches[0, 0, 1, 1]
array([27, 30])
patches[0, 0, 2, 0]
array([36, 45])
patches[0, 0, 2, 1]
array([42, 45])
```

Let us now try to reconstruct the original image from the patches by averaging on overlapping areas

reconstructed = image_reconstruct_from_patches_2d(patches, 4, 4, 3)

np.testing.assert_array_equal(oneimage, reconstructed)

ThePatchExtractor class works in the same way as extractpatches2d only it supports multiple images as input It is implemented as an estimator so it can be used in pipelines See

```
five_images = np.arange(5 * 4 * 4 * 3).reshape(5, 4, 4, 3)
patches = imagePatchExtractor(2, 2).transform(five_images)
patches.shape
(4, 5, 2, 2, 3)
```

Connectivity graph of an image

Several estimators in the scikitlearn can use connectivity information between features or samples For instance Ward clustering Hierarchical clustering can cluster together only neighboring pixels of an image thus forming contiguous patches

For this purpose the estimators use a 'connectivity' matrix giving which samples are connected

scikitlearn user guide Release 0213

The function `imgtograph` returns such a matrix from a 2D or 3D image. Similarly `gridtograph` build a connectivity matrix for images given the shape of these image

These matrices can be used to impose connectivity in estimators that use connectivity information such as Ward clustering. Hierarchical clustering, but also to build precomputed kernels or similarity matrices

Note Examples

- A demo of structured Ward hierarchical clustering on an image of coins
- Spectral clustering for image segmentation
- Feature agglomeration vs univariate selection

353 Preprocessing data

The `sklearn.preprocessing` package provides several common utility functions and transformer classes to change raw feature vectors into a representation that is more suitable for the downstream estimators

In general learning algorithms benefit from standardization of the data set. If some outliers are present in the set, robust scalers or transformers are more appropriate. The behaviors of the different scalers, transformers, and normalizers on

a dataset containing marginal outliers is highlighted in `Compare the effect of different scalers on data with outliers`. Standardization or mean removal and variance scaling

Standardization of datasets is a common requirement for many machine learning estimators implemented in `scikitlearn`. They might behave badly if the individual features do not more or less look like standard normally distributed data: Gaussian with zero mean and unit variance

In practice, we often ignore the shape of the distribution and just transform the data to center it by removing the mean value of each feature, then scale it by dividing nonconstant features by their standard deviation

For instance, many elements used in the objective function of a learning algorithm, such as the RBF (Radial Basis Function) kernel of Support Vector Machines or the $L1$ and $L2$ regularizers of linear models, assume that all features are centered around zero and

have variance in the same order. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected

The function `scale` provides a quick and easy way to perform this operation on a single arraylike dataset

```
from sklearn import preprocessing
```

```
import numpy as np
```

598 Chapter 3 User Guide

scikitlearn user guide Release 0213

Xtrain nparray 1 1 2
2 0 0
0 1 1

Xscaled preprocessingXtrain
Xscaled

array 0 122 133
122 0 026
122 122 106

Scaled data has zero mean and unit variance

Xscaledmeanaxis0
array 0 0 0
Xscaledstdaxis0
array 1 1 1

The preprocessing module further provides a utility class StandardScaler that implements the
Transformer API (Application Programming Interface) to compute the mean and standard deviation on a training set so as to be able to later reap
the same transformation on the testing set This class is hence suitable for use in the early steps of a sklearn

pipeline Pipeline
scaler preprocessingStandardScalerfitXtrain
scaler
StandardScaler copy True withmean True withstd True
scaler mean
array 1 0 033
scaler scale
array 081 081 124
scaler transformXtrain
array 0 122 133
122 0 026
122 122 106

The scaler instance can then be used on new data to transform it the same way it did on the training set

Xtest 1 1 0
scaler transformXtest
array 244 122 026

It is possible to disable either centering or scaling by either passing withmean False or withstd False to
the constructor of StandardScaler

Scaling features to a range

An alternative standardization is scaling features to lie between a given minimum and maximum value often between
zero and one or so that the maximum absolute value of each feature is scaled to unit size This can be achieved using
MinMaxScaler or MaxAbsScaler respectively

The motivation to use this scaling include robustness to very small standard deviations of features and preserving zero
entries in sparse data

Here is an example to scale a toy data matrix to the 0 1 range

35 Dataset transformations 599

scikitlearn user guide Release 0213

```
Xtrain nparray 1 1 2
2 0 0
0 1 1
```

```
minmaxscaler preprocessingMinMaxScaler
Xtrainminmax minmaxscalerfittransformXtrain
Xtrainminmax
array05 0 1
1 05 033333333
0 1 0
```

The same instance of the transformer can then be applied to some new test data unseen during the fit call the same scaling and shifting operations will be applied to be consistent with the transformation performed on the train data

```
Xtest nparray3 1 4
Xtestminmax minmaxscalertransformXtest
Xtestminmax
array15 0 166666667
```

It is possible to introspect the scaler attributes to find about the exact nature of the transformation learned on the training data

```
minmaxscalerscale
array05 05 033
minmaxscalermin
array0 05 033
```

IfMinMaxScaler is given an explicit featurerangemin max the full formula is

```
Xstd X Xminaxis0 Xmaxaxis0 Xminaxis0
Xscaled Xstd max min min
```

MaxAbsScaler works in a very similar fashion but scales in a way that the training data lies within the range 1 by dividing through the largest maximum value in each feature It is meant for data that is already centered at zero or sparse data

Here is how to use the toy data from the previous example with this scaler

```
Xtrain nparray 1 1 2
2 0 0
0 1 1
```

```
maxabsscaler preprocessingMaxAbsScaler
Xtrainmaxabs maxabsscalerfittransformXtrain
Xtrainmaxabs doctest NORMALIZEWHITESPACE
array 05 1 1
1 0 0
0 1 05
```

```
Xtest nparray 3 1 4
Xtestmaxabs maxabsscalertransformXtest
Xtestmaxabs
array15 1 2
maxabsscalerscale
array2 1 2
```

scikitlearn user guide Release 0213

As withscale the module further provides convenience functions minmaxscale andmaxabsscale if you don't want to create an object

Scaling sparse data

Centering sparse data would destroy the sparseness structure in the data and thus rarely is a sensible thing to do However it can make sense to scale sparse inputs especially if features are on different scales

MaxAbsScaler andmaxabsscale were specifically designed for scaling sparse data and are the recommended way to go about this However scale andStandardScaler can accept scipysparse matrices as input as long aswithmeanFalse is explicitly passed to the constructor Otherwise a ValueError will be raised as silently centering would break the sparsity and would often crash the execution by allocating excessive amounts of memory unintentionally RobustScaler cannot be fitted to sparse inputs but you can use the transform method on sparse inputs

Note that the scalers accept both Compressed Sparse Rows and Compressed Sparse Columns format see scipy sparsecsrmatrix andscipysparsecsrmatrix Any other sparse input will be converted to the Compressed Sparse Rows representation To avoid unnecessary memory copies it is recommended to choose the CSR (Compressed Sparse Row) or CSC representation upstream

Finally if the centered data is expected to be small enough explicitly converting the input to an array using the toarray method of sparse matrices is another option

Scaling data with outliers

If your data contains many outliers scaling using the mean and variance of the data is likely to not work very well In these cases you can use robustscale andRobustScaler as dropin replacements instead They use more robust estimates for the center and range of your data

References

Further discussion on the importance of centering and scaling data is available on this FAQ Should I normalize and rescale the data

Scaling vs Whitening

It is sometimes not enough to center and scale the features independently since a downstream model can further make some assumption on the linear independence of the features

To address this issue you can use sklearn.decomposition.PCA withwhitenTrue to further remove the linear correlation across features

Scaling a 1D array

All above functions ie scale minmaxscale maxabsscale androbustscale accept 1D array which can be useful in some specific case

35 Dataset transformations 601

Centering kernel matrices

If you have a kernel matrix of a kernel κ that computes a dot product in a feature space defined by function ϕ h a `KernelCenterer` can transform the kernel matrix so that it contains inner products in the feature space defined by ϕ h followed by removal of the mean in that space

Nonlinear transformation

Two types of transformations are available quantile transforms and power transforms Both quantile and power transforms are based on monotonic transformations of the features and thus preserve the rank of the values along each feature

Quantile transforms put all features into the same desired distribution based on the formula $\Phi^{-1}(\frac{i}{n+1})$ where Φ is the cumulative distribution function of the feature and Φ^{-1} the quantile function of the desired output distribution Φ This formula is using the two following facts i if Φ is a random variable with a continuous cumulative distribution function Φ then $\Phi(\Phi^{-1}(u))$ is uniformly distributed on $[0, 1]$ ii if Φ is a random variable with uniform distribution on $[0, 1]$ then $\Phi^{-1}(\Phi)$ has distribution Φ By performing a rank transformation a quantile transform smooths out unusual distributions and is less influenced by outliers than scaling methods It does however distort correlations and distances within and across features

Power transforms are a family of parametric transformations that aim to map data from any distribution to as close to a Gaussian distribution

Mapping to a Uniform distribution

`QuantileTransformer` and `quantiletransform` provide a nonparametric transformation to map the data to a uniform distribution with values between 0 and 1

```
from sklearn.datasets import loadiris
from sklearn.model_selection import train_test_split
iris = loadiris
X, y = iris.data, iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
quantile_transformer = preprocessing.QuantileTransformer(random_state=0)
X_train_trans = quantile_transformer.fit_transform(X_train)
X_test_trans = quantile_transformer.transform(X_test)
np.percentile(X_train, 0, 25, 50, 75, 100)
array(43. 51. 58. 65. 79.)
```

This feature corresponds to the sepal length in cm Once the quantile transformation applied those landmarks approach closely the percentiles previously defined

```
np.percentile(X_train_trans, 0, 25, 50, 75, 100)
array(0.00 0.24 0.49 0.73 0.99)
This can be confirmed on a independent testing set with similar remarks
np.percentile(X_test, 0, 25, 50, 75, 100)
array(44. 51.25 57.5 61.75 73.)
np.percentile(X_test_trans, 0, 25, 50, 75, 100)
```


Mapping to a Gaussian distribution

In many modeling scenarios normality of the features in a dataset is desirable Power transforms are a family of parametric monotonic transformations that aim to map data from any distribution to as close to a Gaussian distribution as possible in order to stabilize variance and minimize skewness

PowerTransformer currently provides two such power transformations the YeoJohnson transform and the BoxCox transform

The YeoJohnson transform is given by

$$\lambda = \begin{cases} \lambda & \text{if } x \geq 0 \\ 1 - \lambda & \text{if } x < 0 \end{cases}$$
$$\ln \left(\lambda x + 1 \right) \text{ if } x \geq 0$$
$$-\ln \left(-\lambda x + 1 \right) \text{ if } x < 0$$

while the BoxCox transform is given by

$$\lambda = \begin{cases} \lambda & \text{if } x \geq 0 \\ -\lambda & \text{if } x < 0 \end{cases}$$
$$\ln \left(\lambda x + 1 \right) \text{ if } x \geq 0$$
$$\ln \left(-\lambda x + 1 \right) \text{ if } x < 0$$

BoxCox can only be applied to strictly positive data In both methods the transformation is parameterized by λ which is determined through maximum likelihood estimation Here is an example of using BoxCox to map samples drawn from a lognormal distribution to a normal distribution

```
pt = preprocessing.PowerTransformer(method='boxcox', standardize=False)
Xlognormal = np.random.RandomState(616).lognormal(size=(3, 3))
Xlognormal
array([[128. 118.  84.],
       [ 94. 160.  38.],
       [135.  21. 109.]])
pt.fit_transform(Xlognormal)
array([[ 49.  17.  15.],
       [  5.  58.  57.],
       [ 69.  84.  10.]])
```

While the above example sets the standardize option to False PowerTransformer will apply zero mean

unit variance normalization to the transformed output by default

Below are examples of BoxCox and YeoJohnson applied to various probability distributions Note that when applied to certain distributions the power transforms achieve very Gaussianlike results but with others they are ineffective This highlights the importance of visualizing the data before and after transformation

It is also possible to map data to a normal distribution using QuantileTransformer by setting

```
output_distribution = 'normal'
qt = preprocessing.QuantileTransformer(
    output_distribution='normal', random_state=0)
Xtrans = qt.fit_transform(X)
Xtrans
array([[ 4.  2.  1.  0.],
       [44. 22. 11.  0.],
       [44. 22. 12.  0.]])
```


77 41 67 25
77 42 67 25
79 44 69 25

Thus the median of the input becomes the mean of the output centered at 0 The normal output is clipped so that the input's minimum and maximum — corresponding to the 1e7 and 1 - 1e7 quantiles respectively — do not become infinite under the transformation

Normalization

Normalization is the process of scaling individual samples to have unit norm This process can be useful if you plan to use a quadratic form such as the dotproduct or any other kernel to quantify the similarity of any pair of samples This assumption is the base of the Vector Space Model often used in text classification and clustering contexts The function normalize provides a quick and easy way to perform this operation on a single arraylike dataset either using the l1orl2norms

```
X = [[1, 1, 2],
     [2, 0, 0],
     [0, 1, 1]]
Xnormalized = preprocessing.normalize(X, norml2)
Xnormalized
array([0.4, 0.4, 0.8],
      [1, 0, 0],
      [0, 0.7, 0.7])
```

The preprocessing module further provides a utility class Normalizer that implements the same operation using the Transformer API (Application Programming Interface) even though the fit method is useless in this case the class is stateless as this operation treats samples independently

This class is hence suitable for use in the early steps of a sklearn pipeline Pipeline

```
normalizer = preprocessing.Normalizer.fit(X) # fit does nothing
normalizer
```

```
Normalizer(copy=True, norml2=1)
```

The normalizer instance can then be used on sample vectors as any transformer

```
normalizer.transform(X)
array([0.4, 0.4, 0.8],
      [1, 0, 0],
      [0, 0.7, 0.7])
normalizer.transform([1, 1, 0])
array([0.7, 0.7, 0])
```

Sparse input

normalize and Normalizer accept both dense arraylike and sparse matrices from scipy sparse as input For sparse input the data is converted to the Compressed Sparse Rows representation see scipy sparse csr matrix before being fed to efficient Cython routines To avoid unnecessary memory copies it is recommended to choose the CSR (Compressed Sparse Row) representation upstream

scikitlearn user guide Release 0213

Encoding categorical features

Often features are not given as continuous values but categorical For example a person could have features male female from Europe from US from Asia uses Firefox uses Chrome uses Safari uses Internet Explorer Such features can be efficiently coded as integers for instance male from US uses Internet Explorer could be expressed as 0 1 3 while female from Asia uses Chrome would be 1 2 1

To convert categorical features to such integer codes we can use the OrdinalEncoder This estimator transforms each categorical feature to one new feature of integers 0 to ncategories - 1

enc_preprocessing.OrdinalEncoder

X = male from US uses Safari female from Europe uses Firefox

↪

enc.fit(X)

OrdinalEncoder(categories=auto dtype = numpy.float64

enc.transform(female from US uses Safari

array([0, 1, 1])

Such integer representation can however not be used directly with all scikitlearn estimators as these expect continuous input and would interpret the categories as being ordered which is often not desired ie the set of browsers was ordered arbitrarily

Another possibility to convert categorical features to features that can be used with scikitlearn estimators is to use a oneofK also known as onehot or dummy encoding This type of encoding can be obtained with the OneHotEncoder which transforms each categorical feature with ncategories possible values into ncategories binary features with one of them 1 and all others 0

Continuing the example above

enc_preprocessing.OneHotEncoder

X = male from US uses Safari female from Europe uses Firefox

↪

enc.fit(X)

OneHotEncoder(categories=None dtype = numpy.float64 handle_unknown = error

nvalues=None sparse=True

enc.transform(female from US uses Safari

male from Europe uses Safari to array

array([[1, 0, 0, 1, 0, 1],

[0, 1, 1, 0, 0, 1]])

By default the values each feature can take is inferred automatically from the dataset and can be found in the categories attribute

enc.categories

array([female, male], dtype=object) array([from Europe, from US],

dtype=object) array([uses Firefox, uses Safari], dtype=object)

It is possible to specify this explicitly using the parameter categories There are two genders four possible

continents and four web browsers in our dataset

genders = female, male

locations = from Africa, from Asia, from Europe, from US

browsers = uses Chrome, uses Firefox, uses IE, uses Safari

enc_preprocessing.OneHotEncoder(categories=genders, locations, browsers)

Note that for there are missing categorical values for the 2nd and 3rd feature

scikitlearn user guide Release 0213

X male from US uses Safari female from Europe uses Firefox

↪

encfitX

OneHotEncodercategoricalfeaturesNone

categories dropNone

dtype numpyfloat64 handleunknownerror

nvaluesNone sparseTrue

enctransformfemale from Asia uses Chrometoarray

array1 0 0 1 0 0 1 0 0 0

If there is a possibility that the training data might have missing categorical features it can often be

better to specify handleunknownignore instead of setting the categories manually as above

Whenhandleunknownignore is specified and unknown categories are encountered during trans

form no error will be raised but the resulting onehot encoded columns for this feature will be all zeros

handleunknownignore is only supported for onehot encoding

enc preprocessingOneHotEncoderhandleunknownignore

X male from US uses Safari female from Europe uses Firefox

↪

encfitX

OneHotEncodercategoricalfeaturesNone categoriesNone dropNone

dtype numpyfloat64 handleunknownignore

nvaluesNone sparseTrue

enctransformfemale from Asia uses Chrometoarray

array1 0 0 0 0 0

It is also possible to encode each column into ncategories 1 columns instead of ncategories columns

by using the drop parameter This parameter allows the user to specify a category for each feature to be dropped This

is useful to avoid colinearity in the input matrix in some classifiers Such functionality is useful for example when

using nonregularized regression LinearRegression since colinearity would cause the covariance matrix to be

noninvertible When this parameter is not None handleunknown must be set to error

X male from US uses Safari female from Europe uses Firefox

↪

dropenc preprocessingOneHotEncoderdropfirstfitX

dropenccategories

arrayfemale male dtypeobject arrayfrom Europe from US

↪dtypeobject arrayuses Firefox uses Safari dtypeobject

dropenctransformXtoarray

array1 1 1

0 0 0

SeeLoading features from dicts for categorical features that are represented as a dict not as scalars

Discretization

Discretization otherwise known as quantization or binning provides a way to partition continuous features into dis

crete values Certain datasets with continuous features may benefit from discretization because discretization can

transform the dataset of continuous attributes to one with only nominal attributes

Onehot encoded discretized features can make a model more expressive while maintaining interpretability For

instance preprocessing with a discretizer can introduce nonlinearity to linear models

35 Dataset transformations 607

scikitlearn user guide Release 0213

Kbins discretization

KBinsDiscretizer discretizes features into kbins

X nparray 3 5 15

0 6 14

6 3 11

est preprocessingKBinsDiscretizernbins3 2 2 encodeordinalfitX

By default the output is onehot encoded into a sparse matrix See Encoding categorical features and this can be configured with the encode parameter For each feature the bin edges are computed during fit and together with the number of bins they will define the intervals Therefore for the current example these intervals are defined as

• feature 1 $-\infty-1-122\infty$

• feature 2 $-\infty55\infty$

• feature 3 $-\infty1414\infty$

Based on these bin intervals Xis transformed as follows

esttransformX

array 0 1 1

1 1 1

2 0 0

The resulting dataset contains ordinal attributes which can be further used in a sklearnpipelinePipeline

Discretization is similar to constructing histograms for continuous data However histograms focus on counting features which fall into particular bins whereas discretization focuses on assigning feature values to these bins

KBinsDiscretizer implements different binning strategies which can be selected with the strategy parame

ter The 'uniform' strategy uses constantwidth bins The 'quantile' strategy uses the quantiles values to have equally

populated bins in each feature The 'kmeans' strategy defines bins based on a kmeans clustering procedure performed on each feature independently

Examples

•Using KBinsDiscretizer to discretize continuous features

•Feature discretization

•Demonstrating the different strategies of KBinsDiscretizer

Feature binarization

Feature binarization is the process of thresholding numerical features to get boolean values This can be useful for downstream probabilistic estimators that make assumption that the input data is distributed according to a multivariate Bernoulli distribution For instance this is the case for the sklearnneuralnetworkBernoulliRBM

It is also common among the text processing community to use binary feature values probably to simplify the proba bilistic reasoning even if normalized counts aka term frequencies or TFIDF valued features often perform slightly better in practice

As for the Normalizer the utility class Binarizer is meant to be used in the early stages of sklearn pipelinePipeline Thefit method does nothing as each sample is treated independently of others

608 Chapter 3 User Guide

scikitlearn user guide Release 0213

```
X 1 1 2
  2 0 0
  0 1 1
binarizer preprocessingBinarizerfitX fit does nothing
binarizer
BinarizercopyTrue threshold00
binarizertransformX
array1 0 1
1 0 0
0 1 0
```

It is possible to adjust the threshold of the binarizer

```
binarizer preprocessingBinarizerthreshold11
binarizertransformX
array0 0 1
1 0 0
0 0 0
```

As for theStandardScaler andNormalizer classes the preprocessing module provides a companion function

binarize to be used when the transformer API (Application Programming Interface) is not necessary

Note that the Binarizer is similar to the KBinsDiscretizer whenk = 2 and when the bin edge is at the valuethreshold

Sparse input

binarize andBinarizer accept both dense arraylike and sparse matrices from scipysparse as input

For sparse input the data is converted to the Compressed Sparse Rows representation seescipysparse

csrmatrix To avoid unnecessary memory copies it is recommended to choose the CSR (Compressed Sparse Row) representation up stream

Imputation of missing values

Tools for imputing missing values are discussed at Imputation of missing values

Generating polynomial features

Often it's useful to add complexity to the model by considering nonlinear features of the input data A simple and common method to use is polynomial features which can get features' highorder and interaction terms It is implemented

inPolynomialFeatures

```
import numpy as np
from sklearnpreprocessing import PolynomialFeatures
X = np.arange(6).reshape(3, 2)
```

```
X
array0 1
2 3
4 5
poly = PolynomialFeatures2
polyfittransformX
35 Dataset transformations 609
```

scikitlearn user guide Release 0213

```
array 1 0 1 0 0 1
1 2 3 4 6 9
1 4 5 16 20 25
The features of X have been transformed from [1 2]to1 [1 2]
1 [1]2 [2]
2
```

In some cases only interaction terms among features are required and it can be gotten with the setting
interactiononlyTrue

```
X nparange9reshape3 3
X
array0 1 2
3 4 5
6 7 8
poly PolynomialFeaturesdegree3 interactiononly True
polyfittransformX
array 1 0 1 2 0 0 2 0
1 3 4 5 12 15 20 60
1 6 7 8 42 48 56 336
```

The features of X have been transformed from [1 2 3]to1 [1 2 3] [1]2 [1]3 [2]3 [1]2[3]
Note that polynomial features are used implicitly in kernel methods eg sklearnsvmSVC sklearn
decompositionKernelPCA when using polynomial Kernel functions
SeePolynomial interpolation for Ridge regression using created polynomial features

Custom transformers

Often you will want to convert an existing Python function into a transformer to assist in data cleaning or processing
You can implement a transformer from an arbitrary function with FunctionTransformer For example to build
a transformer that applies a log transformation in a pipeline do

```
import numpy as np
from sklearnpreprocessing import FunctionTransformer
transformer FunctionTransformernplog1p validate True
X nparray0 1 2 3
transformertransformX
array0 069314718
109861229 138629436
```

You can ensure that func andinversefunc are the inverse of each other by setting checkinverseTrue
and calling fit beforetransform Please note that a warning is raised and can be turned into an error with a
filterwarnings

```
import warnings
warningsfilterwarningseerror message checkinverse
categoryUserWarning append False
```

For a full code example that demonstrates using a FunctionTransformer to do custom feature selection see

Using FunctionTransformer to select columns

354 Imputation of missing values

For various reasons many real world datasets contain missing values often encoded as blanks NaNs or other place
holders Such datasets however are incompatible with scikitlearn estimators which assume that all values in an array

scikitlearn user guide Release 0213

are numerical and that all have and hold meaning A basic strategy to use incomplete datasets is to discard entire rows and/or columns containing missing values However this comes at the price of losing data which may be valuable even though incomplete A better strategy is to impute the missing values ie to infer them from the known part of the data See the Glossary of Common Terms and API (Application Programming Interface) Elements entry on imputation

Univariate vs Multivariate Imputation
One type of imputation algorithm is univariate which imputes values in the ith feature dimension using only non missing values in that feature dimension eg `imputeSimpleImputer` By contrast multivariate imputation algorithms use the entire set of available feature dimensions to estimate the missing values eg `imputeIterativeImputer`

Univariate feature imputation

The `SimpleImputer` class provides basic strategies for imputing missing values Missing values can be imputed with a provided constant value or using the statistics mean median or most frequent of each column in which the missing values are located This class also allows for different missing values encodings
The following snippet demonstrates how to replace missing values encoded as `np.nan` using the mean value of the columns axis 0 that contain the missing values

```
import numpy as np
from sklearn.impute import SimpleImputer
imp = SimpleImputer(missing_values=np.nan, strategy='mean')
imp.fit(1 2 np.nan 3 7 6)
SimpleImputer(add_indicator=False, copy=True, fill_value=None,
missing_values=np.nan, strategy='mean', verbose=0)
X = np.nan 2 6 np.nan 7 6
print(imp.transform(X))
4 2
6 3 6 6
7 6
```

The `SimpleImputer` class also supports sparse matrices

```
import scipy.sparse as sp
X = sp.cscmatrix(1 2 0 1 8 4)
imp = SimpleImputer(missing_values=1, strategy='mean')
imp.fit(X)
SimpleImputer(add_indicator=False, copy=True, fill_value=None,
missing_values=1, strategy='mean', verbose=0)
X_test = sp.cscmatrix(1 2 6 1 7 6)
print(imp.transform(X_test).toarray())
3 2
6 3
7 6
```

Note that this format is not meant to be used to implicitly store missing values in the matrix because it would densify it at transform time Missing values encoded by 0 must be used with dense input

The `SimpleImputer` class also supports categorical data represented as string values or pandas categoricals when using the `most_frequent` or `constant` strategy

```
import pandas as pd
df = pd.DataFrame({'x':
np.nan, 'y':
35})
Dataset transformations 611
```

scikitlearn user guide Release 0213

a npnan
b y dtypecategory

imp SimpleImputerstrategymostfrequent
printimpfittransformdf
a x
a y
a y
b y

Multivariate feature imputation

A more sophisticated approach is to use the IterativeImputer class which models each feature with missing values as a function of other features and uses that estimate for imputation It does so in an iterated roundrobin fashion at each step a feature column is designated as output yand the other feature columns are treated as inputs X A regressor is fit on X y for knowny Then the regressor is used to predict the missing values of y This is done for each feature in an iterative fashion and then is repeated for maxiter imputation rounds The results of the final imputation round are returned

Note This estimator is still experimental for now the predictions and the API (Application Programming Interface) might change without any deprecation cycle To use it you need to explicitly import enableiterativeimputer

```
import numpy as np
from sklearn.experimental import enableiterativeimputer
from sklearn.impute import IterativeImputer
imp = IterativeImputer(maxiter=10, random_state=0)
imp.fit(1 2 3 6 4 8 npnan 3 7 npnan)
IterativeImputer(add_indicator=False, estimator=None,
imputation_order='ascending', initial_strategy='mean',
max_iter=10, max_value=None, min_value=None,
missing_values=np.nan, n_nearest_features=None,
random_state=0, sample_posterior=False, tol=0.001,
verbose=0)
X_test = npnan 2 6 npnan npnan 6
# the model learns that the second feature is double the first
print(np.round(imp.transform(X_test), 1))
# 2
# 6 12
# 3 6
```

Both SimpleImputer and IterativeImputer can be used in a Pipeline as a way to build a composite estimator that supports imputation See Imputing missing values before building an estimator
Flexibility of IterativeImputer

There are many well established imputation packages in the R data science ecosystem Amelia mi mice miss Forest etc missForest is popular and turns out to be a particular instance of different sequential imputation algorithms that can all be implemented with IterativeImputer by passing in different regressors to be used for predicting missing feature values In the case of missForest this regressor is a Random Forest See sphxglr/autoexamples/plot_iterative_imputer_variants_comparison.py
612 Chapter 3 User Guide

scikitlearn user guide Release 0213

Multiple vs Single Imputation

In the statistics community it is common practice to perform multiple imputations generating for example mseparate imputations for a single feature matrix Each of these mimputations is then put through the subsequent analysis pipeline eg feature engineering clustering regression classification The mfinal analysis results eg heldout validation errors allow the data scientist to obtain understanding of how analytic results may differ as a consequence of the inherent uncertainty caused by the missing values The above practice is called multiple imputation Our implementation of IterativeImputer was inspired by the R MICE package Multivariate Imputation by Chained Equations¹ but differs from it by returning a single imputation instead of multiple imputations However IterativeImputer can also be used for multiple imputations by applying it repeatedly to the same dataset with different random seeds when sampleposterior=True See² chapter 4 for more discussion on multiple vs single imputations

It is still an open problem as to how useful single vs multiple imputation is in the context of prediction and classification when the user is not interested in measuring uncertainty due to missing values

Note that a call to the transform method ofIterativeImputer is not allowed to change the number of samples Therefore multiple imputations cannot be achieved by a single call to transform

References

Marking imputed values

TheMissingIndicator transformer is useful to transform a dataset into corresponding binary matrix indicating the presence of missing values in the dataset This transformation is useful in conjunction with imputation When using imputation preserving the information about which values had been missing can be informative NaN is usually used as the placeholder for missing values However it enforces the data type to be float The parameter missingvalues allows to specify other placeholder such as integer In the following example we will use 1as missing values

```
from sklearnimpute import MissingIndicator
X = nparray1 1 1 3
  4 1 0 1
  8 1 1 0
indicator = MissingIndicatoremissingvalues1
maskmissingvaluesonly = indicatorfittransformX
maskmissingvaluesonly
array True True False
False True True
False True False
```

Thefeatures parameter is used to choose the features for which the mask is constructed By default it is missingonly which returns the imputer mask of the features containing missing values at fit time indicatorfeatures

```
array0 1 3
Thefeatures parameter can be set to all to returned all features whether or not they contain missing values
indicator = MissingIndicatoremissingvalues1 featuresall
maskall = indicatorfittransformX
```

¹Stef van Buuren Karin GroothuisOudshoorn 2011 “mice Multivariate Imputation by Chained Equations in R” Journal of Statistical Software 45 167
²Roderick J A Little and Donald B Rubin 1986 “Statistical Analysis with Missing Data” John Wiley Sons Inc New York NY USA
35 Dataset transformations 613

scikitlearn user guide Release 0213

```
maskall
array True True False False
False True False True
False True False False
indicatorfeatures
array0 1 2 3
```

When using the MissingIndicator in aPipeline be sure to use the FeatureUnion or ColumnTransformer to add the indicator features to the regular features First we obtain the iris dataset and add some missing values to it

```
from sklearn.datasets import loadiris
from sklearn.impute import SimpleImputer MissingIndicator
from sklearn.model_selection import train_test_split
from sklearn.pipeline import FeatureUnion make_pipeline
from sklearn.tree import DecisionTreeClassifier
X y = loadiris(return_Xy=True)
mask = np.random.randint(0, 2, size=X.shape[0]).astype(np.bool)
X[mask] = np.nan
X_train X_test y_train y_test = train_test_split(X, y, test_size=100,
                                                    random_state=0)
```

Now we create a FeatureUnion All features will be imputed using SimpleImputer in order to enable classifiers to work with this data Additionally it adds the the indicator variables from MissingIndicator

```
transformer = FeatureUnion(
    transformer_list=[
        features=SimpleImputer(strategy='mean'),
        indicators=MissingIndicator(),
    ])
transformer.fit(X_train, y_train)
results = transformer.transform(X_test)
resultsshape
100 8
```

Of course we cannot use the transformer to make any predictions We should wrap this in a Pipeline with a classifier eg a DecisionTreeClassifier to be able to make predictions

```
clf = make_pipeline(transformer, DecisionTreeClassifier)
clf.fit(X_train, y_train)
results = clf.predict(X_test)
resultsshape
100
```

355 Unsupervised dimensionality reduction

If your number of features is high it may be useful to reduce it with an unsupervised step prior to supervised steps Many of the Unsupervised learning methods implement a transform method that can be used to reduce the dimensionality Below we discuss two specific examples of this pattern that are heavily used

Pipelining
The unsupervised data reduction and the supervised estimator can be chained in one step See Pipeline chaining estimators

scikitlearn user guide Release 0213

PCA (Principal Component Analysis) principal component analysis
decompositionPCA looks for a combination of features that capture well the variance of the original features
SeeDecomposing signals in components matrix factorization problems

Examples

- Faces recognition example using eigenfaces and SVMs

Random projections

The module randomprojection provides several tools for data reduction by random projections See the relevant section of the documentation Random Projection

Examples

- The JohnsonLindenstrauss bound for embedding with random projections

Feature agglomeration

clusterFeatureAgglomeration applies Hierarchical clustering to group together features that behave similarly

Examples

- Feature agglomeration vs univariate selection
- Feature agglomeration

Feature scaling

Note that if features have very different scaling or statistical properties clusterFeatureAgglomeration may not be able to capture the links between related features Using a preprocessingStandardScaler can be useful in these settings

356 Random Projection

Thesklearnrandomprojection module implements a simple and computationally efficient way to reduce the dimensionality of the data by trading a controlled amount of accuracy as additional variance for faster processing times and smaller model sizes This module implements two types of unstructured random matrix Gaussian random matrix andsparse random matrix

The dimensions and distribution of random projections matrices are controlled so as to preserve the pairwise distances between any two samples of the dataset Thus random projection is a suitable approximation technique for distance based method

35 Dataset transformations 615

scikitlearn user guide Release 0213

References

- Sanjoy Dasgupta 2000 Experiments with random projection In Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence UAI'00 Craig Boutilier and Moisés Goldszmidt Eds Morgan Kaufmann Publishers Inc San Francisco CA USA 143151
- Ella Bingham and Heikki Mannila 2001 Random projection in dimensionality reduction applications to image and text data In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining KDD '01 ACM New York NY USA 245250

The JohnsonLindenstrauss lemma

The main theoretical result behind the efficiency of random projection is the JohnsonLindenstrauss lemma quoting Wikipedia

In mathematics the JohnsonLindenstrauss lemma is a result concerning lowdistortion embeddings of points from highdimensional into lowdimensional Euclidean space The lemma states that a small set of points in a highdimensional space can be embedded into a space of much lower dimension in such a way that distances between the points are nearly preserved The map used for the embedding is at least Lipschitz and can even be taken to be an orthogonal projection

Knowing only the number of samples the sklearnrandomprojection

johnsonlindenstraussmindim estimates conservatively the minimal size of the random subspace to guarantee a bounded distortion introduced by the random projection

```
from sklearnrandomprojection import johnsonlindenstraussmindim
johnsonlindenstraussmindimnsamples1e6 eps05
```

```
663
johnsonlindenstraussmindimnsamples1e6 eps05 01 001
array 663 11841 1112658
```

```
johnsonlindenstraussmindimnsamples1e4 1e5 1e6 eps01
array 7894 9868 11841
```

Example

- See The JohnsonLindenstrauss bound for embedding with random projections for a theoretical explication on the JohnsonLindenstrauss lemma and an empirical validation using sparse random matrices

References

- Sanjoy Dasgupta and Anupam Gupta 1999 An elementary proof of the JohnsonLindenstrauss Lemma Gaussian random projection

The sklearnrandomprojectionGaussianRandomProjection reduces the dimensionality by projecting the original input space on a randomly generated matrix where components are drawn from the following distribution ¶01

¶¶¶¶¶¶¶¶¶¶

Here a small excerpt which illustrates how to use the Gaussian random projection transformer

scikitlearn user guide Release 0213

```
import numpy as np
from sklearn import randomprojection
X = np.random.rand(100, 10000)
transformer = randomprojection.GaussianRandomProjection
X_new = transformer.fit_transform(X)
X_new.shape
(100, 3947)
```

Sparse random projection

The `sklearn.randomprojection.SparseRandomProjection` reduces the dimensionality by projecting the original input space using a sparse random matrix. Sparse random matrices are an alternative to dense Gaussian random projection matrix that guarantees similar embedding quality while being much more memory efficient and allowing faster computation of the projected data. If we define $s = 1/\text{density}$ the elements of the random matrix are drawn from

$$\begin{cases} \sqrt{\frac{2}{s}} & \text{with probability } \frac{1}{2} \\ -\sqrt{\frac{2}{s}} & \text{with probability } \frac{1}{2} \\ 0 & \text{with probability } 1 - \frac{1}{s} \end{cases}$$

where $\frac{1}{s}$ components is the size of the projected subspace. By default the density of non zero elements is set to the minimum density as recommended by Ping Li et al $1/\sqrt{\text{features}}$. Here a small excerpt which illustrates how to use the sparse random projection transformer

```
import numpy as np
from sklearn import randomprojection
X = np.random.rand(100, 10000)
transformer = randomprojection.SparseRandomProjection
X_new = transformer.fit_transform(X)
X_new.shape
(100, 3947)
```

References

- D Achlioptas 2003 Databasefriendly random projections JohnsonLindenstrauss with binary coins Journal of Computer and System Sciences 66 2003 671–687
- Ping Li Trevor J Hastie and Kenneth W Church 2006 Very sparse random projections In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining KDD '06 ACM New York NY USA 287296

357 Kernel Approximation
This submodule contains functions that approximate the feature mappings that correspond to certain kernels as they are used for example in support vector machines see Support Vector Machines. The following feature functions perform nonlinear transformations of the input which can serve as a basis for linear classification or other algorithms. The advantage of using approximate explicit feature maps compared to the kernel trick which makes use of feature maps implicitly is that explicit mappings can be better suited for online learning and can significantly reduce the cost of learning with very large datasets. Standard kernelized SVMs do not scale well to large datasets but using an

scikitlearn user guide Release 0213

approximate kernel map it is possible to use much more efficient linear SVMs In particular the combination of kernel map approximations with SGDClassifier can make nonlinear learning on large datasets possible
Since there has not been much empirical work using approximate embeddings it is advisable to compare results against exact kernel methods when possible
See also

Polynomial regression extending linear models with basis functions for an exact polynomial transformation
Nystroem Method for Kernel Approximation

The Nystroem method as implemented in Nystroem is a general method for lowrank approximations of kernels
It achieves this by essentially subsampling the data on which the kernel is evaluated By default Nystroem uses the rbf kernel but it can use any kernel function or a precomputed kernel matrix The number of samples used which is also the dimensionality of the features computed is given by the parameter ncomponents
Radial Basis Function Kernel

TheRBFSampler constructs an approximate mapping for the radial basis function kernel also known as Random Kitchen Sinks RR2007 This transformation can be used to explicitly model a kernel map prior to applying a linear algorithm for example a linear SVM (Support Vector Machine)

```
from sklearn.kernelapproximation import RBFSampler
from sklearn.linear_model import SGDClassifier
X = [[0,0,1,1,1,0,0,1],
     [0,0,1,1]]
rbf = RBFSampler(gamma=1, random_state=1)
X_features = rbf.fit_transform(X)
clf = SGDClassifier(max_iter=5,
                    alpha=0.0001, average=False, class_weight=None,
                    early_stopping=False, epsilon=0.1, eta=0.001, fit_intercept=True,
                    l1_ratio=0.15, learning_rate='optimal', loss='hinge', max_iter=5,
                    n_iter_no_change=5, n_jobs=None, penalty='l2', power_t=0.5,
                    random_state=None, shuffle=True, tol=0.0001, validation_fraction=0.1,
                    verbose=0, warm_start=False)
clf.fit(X_features, y)
print(clf.score(X_features, y))
```

The mapping relies on a Monte Carlo approximation to the kernel values The fit function performs the Monte Carlo sampling whereas the transform method performs the mapping of the data Because of the inherent randomness of the process results may vary between different calls to the fit function
The fit function takes two arguments ncomponents which is the target dimensionality of the feature transform and gamma the parameter of the RBF kernel A higher ncomponents will result in a better approximation of the kernel and will yield results more similar to those produced by a kernel SVM (Support Vector Machine) Note that “fitting” the feature function does not actually depend on the data given to the fit function Only the dimensionality of the data is used Details on the method can be found in RR2007
For a given value of ncomponents RBFSampler is often less accurate as Nystroem RBFSampler is cheaper to compute though making use of larger feature spaces more efficient
Examples
35 Dataset transformations 619

Fig 39 Comparing an exact RBF (Radial Basis Function) kernel left with the approximation right

•Explicit feature map approximation for RBF (Radial Basis Function) kernels

Additive Chi Squared Kernel

The additive chi squared kernel is a kernel on histograms often used in computer vision

The additive chi squared kernel as used here is given by

$$K(x,y) = \sum_{i=1}^n \frac{1}{2} \left(\frac{x_i}{y_i} + \frac{y_i}{x_i} \right)$$

This is not exactly the same as sklearnmetricsadditivechi2kernel The authors of VZ2010 prefer the version above as it is always positive definite Since the kernel is additive it is possible to treat all components separately for embedding This makes it possible to sample the Fourier transform in regular intervals instead of approximating using Monte Carlo sampling

The classAdditiveChi2Sampler implements this component wise deterministic sampling Each component is sampled $\frac{1}{2}$ times yielding 2×1 dimensions per input dimension the multiple of two stems from the real and complex part of the Fourier transform In the literature $\frac{1}{2}$ is usually chosen to be 1 or 2 transforming the dataset to $2 \times n$ features in the case of $\frac{1}{2} = 2$

The approximate feature map provided by AdditiveChi2Sampler can be combined with the approximate feature map provided by RBFSampler to yield an approximate feature map for the exponentiated chi squared kernel See theVZ2010 for details and VVZ2010 for combination with the RBFSampler

Skewed Chi Squared Kernel

The skewed chi squared kernel is given by

$$K(x,y) = \prod_{i=1}^n \left(\frac{x_i}{y_i} \right)^{\frac{1}{2}} \left(\frac{y_i}{x_i} \right)^{\frac{1}{2}}$$

It has properties that are similar to the exponentiated chi squared kernel often used in computer vision but allows for a simple Monte Carlo approximation of the feature map

scikitlearn user guide Release 0213

The usage of the SkewedChi2Sampler is the same as the usage described above for the RBFSampler. The only difference is in the free parameter that is called γ . For a motivation for this mapping and the mathematical details see LS2010

Mathematical Details

Kernel methods like support vector machines or kernelized PCA (Principal Component Analysis) rely on a property of reproducing kernel Hilbert spaces. For any positive definite kernel function κ a so called Mercer kernel it is guaranteed that there exists a mapping ϕ into a Hilbert space \mathcal{H} such that

$$\kappa(x, y) = \langle \phi(x), \phi(y) \rangle$$

Where $\langle \cdot, \cdot \rangle$ denotes the inner product in the Hilbert space

If an algorithm such as a linear support vector machine or PCA (Principal Component Analysis) relies only on the scalar product of data points x, y one may use the value of $\kappa(x, y)$ which corresponds to applying the algorithm to the mapped data points $\phi(x), \phi(y)$. The advantage of using κ is that the mapping ϕ never has to be calculated explicitly allowing for arbitrary large features even infinite

One drawback of kernel methods is that it might be necessary to store many kernel values $\kappa(x_i, x_j)$ during optimization. If a kernelized classifier is applied to new data x_{new} $\kappa(x_{\text{new}}, x_i)$ needs to be computed to make predictions possibly for many different x_i in the training set.

The classes in this submodule allow to approximate the embedding ϕ thereby working explicitly with the representations $\phi(x)$ which obviates the need to apply the kernel or store training examples.

References

358 Pairwise metrics Affinities and Kernels

The sklearn.metrics.pairwise submodule implements utilities to evaluate pairwise distances or affinity of sets of samples.

This module contains both distance metrics and kernels. A brief summary is given on the two here.

Distance metrics are functions $d(a, b)$ such that $d(a, b) \geq d(a, c)$ if objects a and b are considered “more similar” than objects a and c . Two objects exactly alike would have a distance of zero. One of the most popular examples is Euclidean distance. To be a ‘true’ metric it must obey the following four conditions:

- 1. $d(a, b) \geq 0$ for all a and b .
- 2. $d(a, b) = 0$ if and only if $a = b$ positive definiteness.
- 3. $d(a, b) = d(b, a)$ symmetry.
- 4. $d(a, c) \leq d(a, b) + d(b, c)$ the triangle inequality.

Kernels are measures of similarity i.e. $s(a, b) \geq s(a, c)$ if objects a and b are considered “more similar” than objects a and c . A kernel must also be positive semidefinite.

There are a number of ways to convert between a distance metric and a similarity measure such as a kernel. Let D be the distance and S be the kernel:

$$S = \frac{1}{n \cdot \exp(D)} \quad \text{where one heuristic for choosing } \gamma \text{ is } \gamma = \frac{1}{\sqrt{2 \cdot \text{num_features}}}$$

35 Dataset transformations 621

scikitlearn user guide Release 0213

The distances between the row vectors of X and the row vectors of Y can be evaluated using `pairwise_distances`. If Y is omitted the pairwise distances of the row vectors of X are calculated. Similarly `pairwise_kernels` can be used to calculate the kernel between X and Y using different kernel functions. See the API (Application Programming Interface) reference for more details.

```
import numpy as np
from sklearn.metrics import pairwise_distances
from sklearn.metrics.pairwise import pairwise_kernels
X = np.array([2, 3, 3, 5, 5, 8])
Y = np.array([1, 0, 2, 1])
pairwise_distances(X, Y, metric='manhattan')
array([[4, 2],
       [7, 5],
       [12, 10],
       [3, 0],
       [5, 8],
       [5, 0]])
pairwise_kernels(X, Y, metric='linear')
array([[2, 7],
       [3, 11],
       [5, 18]])
```

Cosine similarity

`cosine_similarity` computes the L2-normalized dot product of vectors. That is, if x and y are row vectors, their cosine similarity is defined as

$$\frac{x \cdot y}{\|x\| \|y\|}$$

This is called cosine similarity because Euclidean L2 normalization projects the vectors onto the unit sphere and their dot product is then the cosine of the angle between the points denoted by the vectors.

This kernel is a popular choice for computing the similarity of documents represented as tfidf vectors.

`cosine_similarity` accepts sparse matrices. Note that the tfidf functionality in `sklearn.feature_extraction.text` can produce normalized vectors in which case `cosine_similarity` is equivalent to `linear_kernel` (only slower).

References

- CD Manning, P. Raghavan and H. Schütze. 2008. Introduction to Information Retrieval. Cambridge University Press. <https://nlp.stanford.edu/IR-book/html/htmledition/the-vector-space-model-for-scoring-1.html>

Linear kernel

The function `linear_kernel` computes the linear kernel, which is a special case of `polynomial_kernel` with `degree=1` and `coef0=0` (homogeneous). If x and y are column vectors, their linear kernel is

$$x \cdot y^T$$

Polynomial kernel

The function `polynomialkernel` computes the degreed polynomial kernel between two vectors The polynomial kernel represents the similarity between two vectors Conceptually the polynomial kernels considers not only the similarity between vectors under the same dimension but also across dimensions When used in machine learning algorithms this allows to account for feature interaction

The polynomial kernel is defined as

$$k(x,y) = (x^T y + c)^d$$

where

- x, y are the input vectors
 - d is the kernel degree
- If $d = 0$ the kernel is said to be homogeneous

Sigmoid kernel

The function `sigmoidkernel` computes the sigmoid kernel between two vectors The sigmoid kernel is also known as hyperbolic tangent or Multilayer Perceptron because in the neural network field it is often used as neuron activation function It is defined as

$$k(x,y) = \tanh(\gamma x^T y + c)$$

where

- x, y are the input vectors
- γ is known as slope
- c is known as intercept

RBF (Radial Basis Function) kernel

The function `rbfkernel` computes the radial basis function RBF (Radial Basis Function) kernel between two vectors This kernel is defined as

$$k(x,y) = \exp(-\gamma \|x - y\|^2)$$

where x and y are the input vectors If $\gamma = 2$ the kernel is known as the Gaussian kernel of variance $\frac{1}{2}$

Laplacian kernel

The function `laplaciankernel` is a variant on the radial basis function kernel defined as

$$k(x,y) = \exp(-\gamma (\|x - y\|_1 + 1))$$

where x and y are the input vectors and $\|x - y\|_1$ is the Manhattan distance between the input vectors

It has proven useful in ML applied to noiseless data See eg Machine learning for quantum mechanics in a nutshell

scikitlearn user guide Release 0213

Chisquared kernel

The chisquared kernel is a very popular choice for training nonlinear SVMs in computer vision applications It can be computed using chi2kernel and then passed to an sklearnsvmSVC withkernelprecomputed

```
from sklearnsvm import SVC (Support Vector Classification)
from sklearnmetricspairwise import chi2kernel
```

```
X 0 1 1 0 2 8 7 3
```

```
y 0 1 0 1
```

```
K chi2kernelX gamma5
```

```
K
```

```
array1 036787944 089483932 058364548
```

```
036787944 1 051341712 083822343
```

```
089483932 051341712 1 07768366
```

```
058364548 083822343 07768366 1
```

```
svm SVCkernelprecomputedfitK y
```

```
svmpredictK
```

```
array0 1 0 1
```

It can also be directly used as the kernel argument

```
svm SVCkernelchi2kernelfitX y
```

```
svmpredictX
```

```
array0 1 0 1
```

The chi squared kernel is given by

$\frac{1}{2} \sum \exp$

$-\frac{1}{2} \sum$

$\frac{1}{2} \sum - \frac{1}{2} \sum$

$\frac{1}{2} \sum$

The data is assumed to be nonnegative and is often normalized to have an L1norm of one The normalization is rationalized with the connection to the chi squared distance which is a distance between discrete probability distributions

The chi squared kernel is most commonly used on histograms bags of visual words

References

- Zhang J and Marszalek M and Lazebnik S and Schmid C Local features and kernels for classification of texture and object categories A comprehensive study International Journal of Computer Vision 2007 <https://research.microsoft.com/en-us/people/manik/projectstradeoffpapers/ZhangIJCV06pdf>

359 Transforming the prediction target y

These are transformers that are not intended to be used on features only on supervised learning targets See also Transforming target in regression if you want to transform the prediction target for learning but evaluate the model in the original untransformed space

Label binarization

LabelBinarizer is a utility class to help create a label indicator matrix from a list of multiclass labels

624 Chapter 3 User Guide

scikitlearn user guide Release 0213

from sklearn import preprocessing

lb_preprocessingLabelBinarizer

lbfit1 2 6 4 2

LabelBinarizer(neglabel=0, poslabel=1, sparseoutput=False)

lbclasses

array1 2 4 6

lbtransform1 6

array1 0 0 0

0 0 0 1

For multiple labels per instance use MultiLabelBinarizer

lb_preprocessingMultiLabelBinarizer

lbfittransform1 2 3

array1 1 0

0 0 1

lbclasses

array1 2 3

Label encoding

LabelEncoder is a utility class to help normalize labels such that they contain only values between 0 and nclasses

1 This is sometimes useful for writing efficient Cython routines LabelEncoder can be used as follows

from sklearn import preprocessing

le_preprocessingLabelEncoder

lefit1 2 2 6

LabelEncoder

leclasses

array1 2 6

letransform1 1 2 6

array0 0 1 2

leinversetransform0 0 1 2

array1 1 2 6

It can also be used to transform nonnumerical labels as long as they are hashable and comparable to numerical

labels

le_preprocessingLabelEncoder

lefitparis paris tokyo amsterdam

LabelEncoder

listleclasses

amsterdam paris tokyo

letransformtokyo tokyo paris

array2 2 1

listleinversetransform2 2 1

tokyo tokyo paris

36 Dataset loading utilities

The sklearn datasets package embeds some small toy datasets as introduced in the Getting Started section

This package also features helpers to fetch larger datasets commonly used by the machine learning community to

benchmark algorithms on data that comes from the ‘real world’

36 Dataset loading utilities 625

scikitlearn user guide Release 0213

To evaluate the impact of the scale of the dataset `nsamples` and `nfeatures` while controlling the statistical properties of the data typically the correlation and informativeness of the features it is also possible to generate synthetic data

361 General dataset API (Application Programming Interface)

There are three main kinds of dataset interfaces that can be used to get datasets depending on the desired type of dataset

The dataset loaders They can be used to load small standard datasets described in the Toy datasets section

The dataset fetchers They can be used to download and load larger datasets described in the Real world datasets section

Both loaders and fetchers functions return a dictionarylike object holding at least two items an array of shape `nsamples nfeatures` with key `data` except for `20newsgroups` and a numpy array of length `nsamples` containing the target values with key `target`

It's also possible for almost all of these function to constrain the output to be a tuple containing only the data and the target by setting the `returnXy` parameter to `True`

The datasets also contain a full description in their `DESCR` attribute and some contain `featurenames` and `targetnames` See the dataset descriptions below for details

The dataset generation functions They can be used to generate controlled synthetic datasets described in the Generated datasets section

These functions return a tuple `X y` consisting of a `nsamples nfeatures` numpy array `X` and an array of length `nsamples` containing the targets `y`

In addition there are also miscellaneous tools to load datasets of other formats or from other locations described in the Loading other datasets section

362 Toy datasets

scikitlearn comes with a few small standard datasets that do not require to download any file from some external website

They can be loaded using the following functions

`loadboston` `returnXy` Load and return the boston houseprices dataset regression

`loadiris` `returnXy` Load and return the iris dataset classification

`loaddiabetes` `returnXy` Load and return the diabetes dataset regression

`loaddigits` `nclass` `returnXy` Load and return the digits dataset classification

`loadlinnerud` `returnXy` Load and return the linnerud dataset multivariate regression

`loadwine` `returnXy` Load and return the wine dataset classification

`loadbreastcancer` `returnXy` Load and return the breast cancer wisconsin dataset classification

These datasets are useful to quickly illustrate the behavior of the various algorithms implemented in scikitlearn They are however often too small to be representative of real world machine learning tasks

scikitlearn user guide Release 0213

Boston house prices dataset

Data Set Characteristics

Number of Instances 506

Number of Attributes 13 numericcategorical predictive Median Value attribute 14 is usually the target

Attribute Information in order

- CRIM per capita crime rate by town
- ZN proportion of residential land zoned for lots over 25000 sqft
- INDUS proportion of nonretail business acres per town
- CHAS Charles River dummy variable 1 if tract bounds river 0 otherwise
- NOX nitric oxides concentration parts per 10 million
- RM average number of rooms per dwelling
- AGE proportion of owneroccupied units built prior to 1940
- DIS weighted distances to five Boston employment centres
- RAD index of accessibility to radial highways
- TAX fullvalue propertytax rate per 10000
- PTRATIO pupilteacher ratio by town
- B 1000Bk 0632 where Bk is the proportion of blacks by town
- LSTAT lower status of the population
- MEDV Median value of owneroccupied homes in 1000's

Missing Attribute Values None

Creator Harrison D and Rubinfeld DL

This is a copy of UCI ML housing dataset <https://archive.ics.uc.edu/ml/machine-learning-databases/housing>

This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University

The Boston houseprice data of Harrison D and Rubinfeld DL 'Hedonic prices and the demand for clean air' J

Environ Economics Management vol5 81102 1978 Used in Belsley Kuh Welsch 'Regression diagnostics

' Wiley 1980 NB Various transformations are used in the table on pages 244261 of the latter

The Boston houseprice data has been used in many machine learning papers that address regression problems

References

- Belsley Kuh Welsch 'Regression diagnostics Identifying Influential Data and Sources of Collinearity' Wiley 1980 244261
- QuinlanR 1993 Combining InstanceBased and ModelBased Learning In Proceedings on the Tenth International Conference of Machine Learning 236243 University of Massachusetts Amherst Morgan Kaufmann

36 Dataset loading utilities 627

scikitlearn user guide Release 0213

Iris plants dataset

Data Set Characteristics

Number of Instances 150 50 in each of three classes

Number of Attributes 4 numeric predictive attributes and the class

Attribute Information

- sepal length in cm
- sepal width in cm
- petal length in cm
- petal width in cm
- class

-IrisSetosa

-IrisVersicolour

-IrisVirginica

Summary Statistics

sepal length 43 79 584 083 07826

sepal width 20 44 305 043 04194

petal length 10 69 376 176 09490 high

petal width 01 25 120 076 09565 high

Missing Attribute Values None

Class Distribution 333 for each of 3 classes

Creator RA Fisher

Donor Michael Marshall MARSHALLPLUioarcnasagov

Date July 1988

The famous Iris database first used by Sir RA Fisher The dataset is taken from Fisher's paper Note that it's the

same as in R but not as in the UCI Machine Learning Repository which has two wrong data points

This is perhaps the best known database to be found in the pattern recognition literature Fisher's paper is a classic in

the field and is referenced frequently to this day See Duda Hart for example The data set contains 3 classes of

50 instances each where each class refers to a type of iris plant One class is linearly separable from the other 2 the

latter are NOT linearly separable from each other

References

• Fisher RA "The use of multiple measurements in taxonomic problems" Annual Eugenics 7 Part II 179188

1936 also in "Contributions to Mathematical Statistics" John Wiley NY 1950

• Duda RO Hart PE 1973 Pattern Classification and Scene Analysis Q327D83 John Wiley Sons

ISBN 0471223611 See page 218

• Dasarathy BV 1980 "Nosing Around the Neighborhood A New System Structure and Classification Rule

for Recognition in Partially Exposed Environments" IEEE Transactions on Pattern Analysis and Machine

Intelligence V ol PAMI2 No 1 6771

628 Chapter 3 User Guide

scikitlearn user guide Release 0213

- Gates GW 1972 “The Reduced Nearest Neighbor Rule” IEEE Transactions on Information Theory May 1972 431433

- See also 1988 MLC Proceedings 5464 Cheeseman et al”s AUTOCLASS II conceptual clustering system

finds 3 classes in the data

- Many many more

Diabetes dataset

Ten baseline variables age sex body mass index average blood pressure and six blood serum measurements were obtained for each of n 442 diabetes patients as well as the response of interest a quantitative measure of disease progression one year after baseline

Data Set Characteristics

Number of Instances 442

Number of Attributes First 10 columns are numeric predictive values

Target Column 11 is a quantitative measure of disease progression one year after baseline

Attribute Information

- Age
- Sex
- Body mass index
- Average blood pressure
- S1
- S2
- S3
- S4
- S5
- S6

Note Each of these 10 feature variables have been mean centered and scaled by the standard deviation times nsamples ie the sum of squares of each column totals 1

Source URL <https://www4.stat.ncsu.edu/boos/varselect/diabetes.html>

For more information see Bradley Efron Trevor Hastie Iain Johnstone and Robert Tibshirani 2004 “Least Angle Regression” Annals of Statistics with discussion 407499 https://web.stanford.edu/hastie/Papers/LARS_LeastAngle2002.pdf

Optical recognition of handwritten digits dataset

Data Set Characteristics

Number of Instances 5620

Number of Attributes 64

Attribute Information 8x8 image of integer pixels in the range 016

36 Dataset loading utilities 629

scikitlearn user guide Release 0213

Missing Attribute Values None

Creator

5 Alpaydin alpaydin " bounedutr

Date July 1998

This is a copy of the test set of the UCI ML handwritten digits datasets <https://archive.ics.uci.edu/ml/datasets/Optical>

Recognition of Handwritten Digits

The data set contains images of handwritten digits 10 classes where each class refers to a digit

Preprocessing programs made available by NIST were used to extract normalized bitmaps of handwritten digits from a preprinted form From a total of 43 people 30 contributed to the training set and different 13 to the test set 32x32 bitmaps are divided into nonoverlapping blocks of 4x4 and the number of on pixels are counted in each block This generates an input matrix of 8x8 where each element is an integer in the range 016 This reduces dimensionality and gives invariance to small distortions

For info on NIST preprocessing routines see M D Garris J L Blue G T Candela D L Dimmick J Geist P J

Grother S A Janet and C L Wilson NIST FormBased Handprint Recognition System NISTIR 5469 1994

References

- C Kaynak 1995 Methods of Combining Multiple Classifiers and Their Applications to Handwritten Digit Recognition MSc Thesis Institute of Graduate Studies in Science and Engineering Bogazici University
- 5 Alpaydin C Kaynak 1998 Cascading Classifiers Kybernetika
- Ken Tang and Ponnuthurai N Suganthan and Xi Yao and A Kai Qin Linear dimensionality reduction using relevance weighted LDA School of Electrical and Electronic Engineering Nanyang Technological University 2005
- Claudio Gentile A New Approximate Maximal Margin Classification Algorithm NIPS 2000

Linnerrud dataset

Data Set Characteristics

Number of Instances 20

Number of Attributes 3

Missing Attribute Values None

The Linnerud dataset contains two small datasets

- physiological CSV containing 20 observations on 3 exercise variables Weight Waist and Pulse
- exercise CSV containing 20 observations on 3 physiological variables Chins Situps and Jumps

References

- Tenenhaus M 1998 La regression PLS theorie et pratique Paris Editions Technic

Wine recognition dataset

Data Set Characteristics

630 Chapter 3 User Guide

scikitlearn user guide Release 0213
Number of Instances 178 50 in each of three classes
Number of Attributes 13 numeric predictive attributes and the class
Attribute Information
• Alcohol
• Malic acid
• Ash
• Alcalinity of ash
• Magnesium
• Total phenols
• Flavanoids
• Nonflavanoid phenols
• Proanthocyanins
• Color intensity
• Hue
• OD280OD315 of diluted wines
• Proline
•class
-class0
-class1
-class2
Summary Statistics
Alcohol 110 148 130 08
Malic Acid 074 580 234 112
Ash 136 323 236 027
Alcalinity of Ash 106 300 195 33
Magnesium 700 1620 997 143
Total Phenols 098 388 229 063
Flavanoids 034 508 203 100
Nonflavanoid Phenols 013 066 036 012
Proanthocyanins 041 358 159 057
Colour Intensity 13 130 51 23
Hue 048 171 096 023
OD280OD315 of diluted wines 127 400 261 071
Proline 278 1680 746 315
Missing Attribute Values None
Class Distribution class0 59 class1 71 class2 48
Creator RA Fisher
Donor Michael Marshall MARSHALLPLUioarcnasagov
36 Dataset loading utilities 631

scikitlearn user guide Release 0213

Date July 1988

This is a copy of UCI ML Wine recognition datasets <https://archive.ics.uc.edu/ml/machine-learning-databases/wine/wine.data>

The data is the results of a chemical analysis of wines grown in the same region in Italy by three different cultivators. There are thirteen different measurements taken for different constituents found in the three types of wine.

Original Owners

Forina M et al PARVUS: An Extendible Package for Data Exploration, Classification and Correlation. Institute of Pharmaceutical and Food Analysis and Technologies, Via Brigata Salerno 16147 Genoa, Italy.

Citation

Lichman M. 2013. UCI Machine Learning Repository. <https://archive.ics.uc.edu/ml/> Irvine, CA: University of California, School of Information and Computer Science.

References

1. S. Aeberhard, D. Coomans and O. de Vel: Comparison of Classifiers in High Dimensional Settings. Tech Rep no 9202, 1992. Dept of Computer Science and Dept of Mathematics and Statistics, James Cook University of North Queensland. Also submitted to Technometrics.

The data was used with many others for comparing various classifiers. The classes are separable though only RDA has achieved 100% correct classification. RDA: 100%, QDA: 99%, LDA: 98%, 1NN: 96%, z-transformed data: All results using the leave-one-out technique.

2. S. Aeberhard, D. Coomans and O. de Vel: "THE CLASSIFICATION PERFORMANCE OF RDA". Tech Rep no 9201, 1992. Dept of Computer Science and Dept of Mathematics and Statistics, James Cook University of North Queensland. Also submitted to Journal of Chemometrics.

Breast cancer wisconsin diagnostic dataset

Data Set Characteristics

Number of Instances: 569

Number of Attributes: 30 numeric predictive attributes and the class

Attribute Information

- radius: mean of distances from center to points on the perimeter
- texture: standard deviation of grayscale values
- perimeter
- area
- smoothness: local variation in radius lengths
- compactness: $\frac{\text{perimeter}^2}{\text{area}}$
- concavity: severity of concave portions of the contour
- concave points: number of concave portions of the contour
- symmetry
- fractal dimension: "coastline approximation" 1

632 Chapter 3 User Guide

scikitlearn user guide Release 0213

The mean standard error and “worst” or largest mean of the three largest values of these features were computed for each image resulting in 30 features For instance field 3 is Mean Radius field 13 is Radius SE field 23 is Worst Radius

•class
-WDBCMalignant
-WDBCBenign
Summary Statistics
radius mean 6981 2811
texture mean 971 3928
perimeter mean 4379 1885
area mean 1435 25010
smoothness mean 0053 0163
compactness mean 0019 0345
concavity mean 00 0427
concave points mean 00 0201
symmetry mean 0106 0304
fractal dimension mean 005 0097
radius standard error 0112 2873
texture standard error 036 4885
perimeter standard error 0757 2198
area standard error 6802 5422
smoothness standard error 0002 0031
compactness standard error 0002 0135
concavity standard error 00 0396
concave points standard error 00 0053
symmetry standard error 0008 0079
fractal dimension standard error 0001 003
radius worst 793 3604
texture worst 1202 4954
perimeter worst 5041 2512
area worst 1852 42540
smoothness worst 0071 0223
compactness worst 0027 1058
concavity worst 00 1252
concave points worst 00 0291
symmetry worst 0156 0664
fractal dimension worst 0055 0208
Missing Attribute Values None
Class Distribution 212 Malignant 357 Benign
Creator Dr William H Wolberg W Nick Street Olvi L Mangasarian
Donor Nick Street
Date November 1995
This is a copy of UCI ML Breast Cancer Wisconsin Diagnostic datasets <https://goo.gl/U2Uwz2>
Features are computed from a digitized image of a fine needle aspirate FNA of a breast mass They describe characteristics of the cell nuclei present in the image
36 Dataset loading utilities 633

scikitlearn user guide Release 0213

Separating plane described above was obtained using Multisurface MethodTree MSMT K P Bennett “Decision Tree Construction Via Linear Programming” Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society pp 97101 1992 a classification method which uses linear programming to construct a decision tree Relevant features were selected using an exhaustive search in the space of 14 features and 13 separating planes The actual linear program used to obtain the separating plane in the 3dimensional space is that described in K P Bennett and O L Mangasarian “Robust Linear Programming Discrimination of Two Linearly Inseparable Sets” Optimization Methods and Software 1 1992 2334

This database is also available through the UW CS ftp server

ftp ftpcswiscedu cd mathprogcpodatasetmachinelearnWDBC

References

- WN Street WH Wolberg and OL Mangasarian Nuclear feature extraction for breast tumor diagnosis ISTSPIE 1993 International Symposium on Electronic Imaging Science and Technology volume 1905 pages 861870 San Jose CA 1993
- OL Mangasarian WN Street and WH Wolberg Breast cancer diagnosis and prognosis via linear program ming Operations Research 434 pages 570577 JulyAugust 1995
- WH Wolberg WN Street and OL Mangasarian Machine learning techniques to diagnose breast cancer from fineneedle aspirates Cancer Letters 77 1994 163171

363 Real world datasets

scikitlearn provides tools to load larger datasets downloading them if necessary

They can be loaded using the following functions

fetcholivettifaces datahome shuffle Load the Olivetti faces dataset from ATT classification

fetch20newsgroups datahome subset Load the filenames and data from the 20 newsgroups

dataset classification

fetch20newsgroupsvectorized subset Load the 20 newsgroups dataset and vectorize it into token

counts classification

fetchlfwpeople datahome funneled Load the Labeled Faces in the Wild LFW people dataset

classification

fetchlfwpairs subset datahome Load the Labeled Faces in the Wild LFW pairs dataset

classification

fetchcovtype datahome Load the covertype dataset classification

fetchrcv1 datahome subset Load the RCV1 multilabel dataset classification

fetchkddcup99 subset datahome shuffle Load the kddcup99 dataset classification

fetchcaliforniahousing datahome Load the California housing dataset regression

The Olivetti faces dataset

This dataset contains a set of face images taken between April 1992 and April 1994 at ATT Laboratories Cam bridge The sklearndatasetsfetcholivettifaces function is the data fetching caching function that downloads the data archive from ATT

As described on the original website

There are ten different images of each of 40 distinct subjects For some subjects the images were taken

634 Chapter 3 User Guide

scikitlearn user guide Release 0213

at different times varying the lighting facial expressions open closed eyes smiling not smiling and facial details glasses no glasses All the images were taken against a dark homogeneous background with the subjects in an upright frontal position with tolerance for some side movement

Data Set Characteristics

Classes 40

Samples total 400

Dimensionality 4096

Features real between 0 and 1

The image is quantized to 256 grey levels and stored as unsigned 8bit integers the loader will convert these to floating point values on the interval 0 1 which are easier to work with for many algorithms

The “target” for this database is an integer from 0 to 39 indicating the identity of the person pictured however with only 10 examples per class this relatively small dataset is more interesting from an unsupervised or semisupervised perspective

The original dataset consisted of 92 x 112 while the version available here consists of 64x64 images

When using these images please give credit to ATT Laboratories Cambridge

The 20 newsgroups text dataset

The 20 newsgroups dataset comprises around 18000 newsgroups posts on 20 topics split in two subsets one for training or development and the other one for testing or for performance evaluation The split between the train and test set is based upon a messages posted before and after a specific date

This module contains two loaders The first one sklearndatasetsfetch20newsgroups returns a list of the raw texts that can be fed to text feature extractors such as sklearnfeatureextractiontext CountVectorizer with custom parameters so as to extract feature vectors The second one sklearn datasetsfetch20newsgroupsvectorized returns readytouse features ie it is not necessary to use a feature extractor

Data Set Characteristics

Classes 20

Samples total 18846

Dimensionality 1

Features text

Usage

Thesklearndatasetsfetch20newsgroups function is a data fetching caching functions that down loads the data archive from the original 20 newsgroups website extracts the archive contents in the scikitlearndata20newshome folder and calls the sklearndatasetsloadfiles on either the training or testing set folder or both of them

```
from sklearndatasets import fetch20newsgroups
```

```
newsgroupstrain fetch20newsgroupssubsettrain
```

```
from pprint import pprint
```

```
pprintlistnewsgroupstraintargetnames
```

```
altatheism
```

```
compgraphics
```

36 Dataset loading utilities 635

scikitlearn user guide Release 0213
composmswindowsmisc
compsysibmpchardware
compsysmachardware
compwindowsex
miscforsale
recautos
recmotorcycles
recsportbaseball
recsportthockey
scicrypt
scieletronics
scimed
scispace
socreligionchristian
talkpoliticsguns
talkpoliticsmideast
talkpoliticsmisc
talkreligionmisc
The real data lies in the filenames andtarget attributes The target attribute is the integer index of the category
newsgroupstrainfilenamesshape
11314
newsgroupstraintargetshape
11314
newsgroupstraintarget10
array 7 4 4 1 14 16 13 3 2 4
It is possible to load only a subselection of the categories by passing the list of the categories to load to the sklearn
datasetsfetch20newsgroups function
cats altatheism scispace
newsgroupstrain fetch20newsgroupssubsettrain categoriescats
listnewsgroupstraintargetnames
altatheism scispace
newsgroupstrainfilenamesshape
1073
newsgroupstraintargetshape
1073
newsgroupstraintarget10
array0 1 1 1 0 1 1 0 0 0
Converting text to vectors
In order to feed predictive or clustering models with the text data one first need to turn the text into vectors
of numerical values suitable for statistical analysis This can be achieved with the utilities of the sklearn
featureextractiontext as demonstrated in the following example that extract TFIDF vectors of unigram
tokens from a subset of 20news
from sklearnfeatureextractiontext import TfidfVectorizer
categories altatheism talkreligionmisc
compgraphics scispace
newsgroupstrain fetch20newsgroupssubsettrain
categoriescategories
vectorizer TfidfVectorizer
636 Chapter 3 User Guide

scikitlearn user guide Release 0213

vectors vectorizerfittransformnewsgroupstraindata

vectorsshape

2034 34118

The extracted TFIDF vectors are very sparse with an average of 159 nonzero components by sample in a more than

30000dimensional space less than 5 nonzero features

vectorsnnp floatvectorsshape0

15901327

sklearn.datasets.fetch_20newsgroups_vectorized is a function which returns ready to use token

counts features instead of file names

Filtering text for more realistic training

It is easy for a classifier to overfit on particular things that appear in the 20 Newsgroups data such as newsgroup

headers Many classifiers achieve very high Fscores but their results would not generalize to other documents that

aren't from this window of time

For example let's look at the results of a multinomial Naive Bayes classifier which is fast to train and achieves a

decent Fscore

```
from sklearn.naive_bayes import MultinomialNB
```

```
from sklearn import metrics
```

```
newsgroup_test, fetch_20newsgroup_subset_test
```

```
categories, categories
```

```
vector_test, vectorizer, transform_newsgroup_test_data
```

```
clf = MultinomialNB(alpha=0.1)
```

```
clf.fit(vector_test, newsgroup_strain_target)
```

```
MultinomialNB(alpha=0.01, class_prior=None, fit_prior=True)
```

```
pred = clf.predict(vector_test)
```

```
metrics.f1_score(newsgroup_test_target, pred, average='macro')
```

0.88213

The example Classification of text documents using sparse features shuffles the training and test data instead of

segmenting by time and in that case multinomial Naive Bayes gets a much higher Fscore of 0.88 Are you suspicious

yet of what's going on inside this classifier

Let's take a look at what the most informative features are

```
import numpy as np
```

```
def show_top10_classifier(vectorizer, categories)
```

```
feature_names = np.asarray(vectorizer.get_feature_names())
```

```
for i, category in enumerate(categories)
```

```
top10 = np.argsort(classifier.coef_[0])
```

```
print('ss', category, join(feature_names[top10])
```

```
show_top10_clf(vectorizer, newsgroup_strain_target_names)
```

```
altatheism.edu it and in you that is of to the
```

```
compgraphics.edu in graphics it is for and of to the
```

```
scispace.edu it that is in and space to of the
```

```
talkreligionmisc not it you in is that and to of the
```

You can now see many things that these features have overfit to

36 Dataset loading utilities 637

scikitlearn user guide Release 0213

- Almost every group is distinguished by whether headers such as NNTPPostingHost and Distribution appear more or less often
- Another significant feature involves whether the sender is affiliated with a university as indicated either by their headers or their signature
- The word "article" is a significant feature based on how often people quote previous posts like this "In article article ID name email address wrote"
- Other features match the names and email addresses of particular people who were posting at the time

With such an abundance of clues that distinguish newsgroups the classifiers barely have to identify topics from text at all and they all perform at the same high level

For this reason the functions that load 20 Newsgroups data provide a parameter called remove telling it what kinds of information to strip out of each file remove should be a tuple containing any subset of headers footers quotes telling it to remove headers signature blocks and quotation blocks respectively

```
newsgroupstest fetch20newsgroupssubsettest
removeheaders footers quotes
categoriescategories
vectorstest vectorizertransformnewsgroupstestdata
pred clfpredictvectorstest
metricsf1scorepred newsgroupstesttarget averagemacro
077310
```

This classifier lost over a lot of its Fscore just because we removed metadata that has little to do with topic classification It loses even more if we also strip this metadata from the training data

```
newsgroupstrain fetch20newsgroupssubsettrain
removeheaders footers quotes
categoriescategories
vectors vectorizerfittransformnewsgroupstraindata
clf MultinomialNBalpha01
clffitvectors newsgroupstraintarget
MultinomialNBalpha001 classpriorNone fitpriorTrue
vectorstest vectorizertransformnewsgroupstestdata
pred clfpredictvectorstest
metricsf1scorenewsgroupstesttarget pred averagemacro
076995
```

Some other classifiers cope better with this harder version of the task Try running Sample pipeline for text feature extraction and evaluation with and without the filter option to compare the results

Recommendation

When evaluating text classifiers on the 20 Newsgroups data you should strip newsgrouprelated metadata In scikitlearn you can do this by setting removeheaders footers quotes The Fscore will be lower because it is more realistic

Examples

- Sample pipeline for text feature extraction and evaluation
- Classification of text documents using sparse features

scikitlearn user guide Release 0213

The Labeled Faces in the Wild face recognition dataset

This dataset is a collection of JPEG pictures of famous people collected over the internet all details are available on the official website

httpviswwwwcsumassedulfw

Each picture is centered on a single face The typical task is called Face Verification given a pair of two pictures a binary classifier must predict whether the two images are from the same person

An alternative task Face Recognition or Face Identification is given the picture of the face of an unknown person identify the name of the person by referring to a gallery of previously seen pictures of identified persons

Both Face Verification and Face Recognition are tasks that are typically performed on the output of a model trained to perform Face Detection The most popular model for Face Detection is called ViolaJones and is implemented in the OpenCV library The LFW faces were extracted by this face detector from various online websites

Data Set Characteristics

Classes 5749

Samples total 13233

Dimensionality 5828

Features real between 0 and 255

Usage

scikitlearn provides two loaders that will automatically download cache parse the metadata files decode the jpeg and convert the interesting slices into memmapped numpy arrays This dataset size is more than 200 MB The first load typically takes more than a couple of minutes to fully decode the relevant part of the JPEG files into numpy arrays If the dataset has been loaded once the following times the loading times less than 200ms by using a memmapped version memoized on the disk in the scikitlearndatalfwhome folder using joblib

The first loader is used for the Face Identification task a multiclass classification task hence supervised learning

```
from sklearndatasets import fetchlfwpeople
lfwpeople fetchlfwpeopleminfacesperperson70 resize04
for nameinlfwpeopletargetnames
    printname
```

Ariel Sharon

Colin Powell

Donald Rumsfeld

George W Bush

Gerhard Schroeder

Hugo Chavez

Tony Blair

The default slice is a rectangular shape around the face removing most of the background

lfwpeopledatatype

dtypefloat32

lfwpeopledatashape

1288 1850

36 Dataset loading utilities 639

scikitlearn user guide Release 0213

lfwpeopleimageesshape

1288 50 37

Each of the 1140 faces is assigned to a single person id in the target array

lfwpeopletargetshape

1288

listlfwpeopletarget10

5 6 3 1 0 1 3 4 3 0

The second loader is typically used for the face verification task each sample is a pair of two picture belonging or not

to the same person

from sklearn.datasets import fetchlfwpairs

lfwpairstrain fetchlfwpairssubsettrain

listlfwpairstraintargetnames

Different persons Same person

lfwpairstrainpairsshape

2200 2 62 47

lfwpairstraindatashape

2200 5828

lfwpairstraintargetshape

2200

Both for the sklearn.datasets.fetchlfwpeople andsklearn.datasets.fetchlfwpairs

function it is possible to get an additional dimension with the RGB color channels by passing colorTrue in

that case the shape will be 2200 2 62 47 3

Thesklearn.datasets.fetchlfwpairs datasets is subdivided into 3 subsets the development train

set the development test set and an evaluation 10folds set meant to compute performance metrics using a

10folds cross validation scheme

References

• Labeled Faces in the Wild A Database for Studying Face Recognition in Unconstrained Environments Gary

B Huang Manu Ramesh Tamara Berg and Erik LearnedMiller University of Massachusetts Amherst

Technical Report 0749 October 2007

Examples

Faces recognition example using eigenfaces and SVMs

Forest covertypes

The samples in this dataset correspond to 30×30m patches of forest in the US collected for the task of predicting

each patch’s cover type ie the dominant species of tree There are seven covertypes making this a multiclass

classification problem Each sample has 54 features described on the dataset’s homepage Some of the features are

boolean indicators while others are discrete or continuous measurements

640 Chapter 3 User Guide

scikitlearn user guide Release 0213

Data Set Characteristics

Classes 7

Samples total 581012

Dimensionality 54

Features int

sklearn.datasets.fetchcovtype will load the covtype dataset it returns a dictionarylike object with the feature matrix in the data member and the target values in target The dataset will be downloaded from the web if necessary

RCV1 dataset

Reuters Corpus Volume I RCV1 is an archive of over 800000 manually categorized newswire stories made available by Reuters Ltd for research purposes The dataset is extensively described in1

Data Set Characteristics

Classes 103

Samples total 804414

Dimensionality 47236

Features real between 0 and 1

sklearn.datasets.fetchrcv1 will load the following version RCV1v2 vectors full sets topics multilabels

from sklearn.datasets import fetchrcv1

rcv1 fetchrcv1

It returns a dictionarylike object with the following attributes

data The feature matrix is a scipy CSR (Compressed Sparse Row) sparse matrix with 804414 samples and 47236 features Nonzero values contains cosinenormalized log TFIDF vectors A nearly chronological split is proposed in1 The first 23149 samples are the training set The last 781265 samples are the testing set This follows the official LYRL2004 chronological split The array has 016 of non zero values

rcv1datashape

804414 47236

target The target values are stored in a scipy CSR (Compressed Sparse Row) sparse matrix with 804414 samples and 103 categories Each sample has a value of 1 in its categories and 0 in others The array has 315 of non zero values

rcv1targetshape

804414 103

sampleid Each sample can be identified by its ID ranging with gaps from 2286 to 810596

rcv1sampleid3

array2286 2287 2288 dtypeuint32

1Lewis D D Yang Y Rose T G Li F 2004 RCV1 A new benchmark collection for text categorization research The Journal of Machine Learning Research 5 361397

36 Dataset loading utilities 641

scikitlearn user guide Release 0213

targetnames The target values are the topics of each sample Each sample belongs to at least one topic and to up to 17 topics There are 103 topics each represented by a string Their corpus frequencies span five orders of magnitude from 5 occurrences for 'GMIL' to 381327 for 'CCAT'

rcv1targetnames3tolist

E11 ECAT M11

The dataset will be downloaded from the rcv1 homepage if necessary The compressed size is about 656 MB

References

Kddcup 99 dataset

The KDD Cup '99 dataset was created by processing the tcpdump portions of the 1998 DARPA Intrusion Detection System IDS Evaluation dataset created by MIT Lincoln Lab 1 The artificial data described on the dataset's

homepage was generated using a closed network and handinjected attacks to produce a large number of different

types of attack with normal activity in the background As the initial goal was to produce a large training set for

supervised learning algorithms there is a large proportion 801 of abnormal data which is unrealistic in real world

and inappropriate for unsupervised anomaly detection which aims at detecting 'abnormal' data ie

1 qualitatively different from normal data

2 in large minority among the observations

We thus transform the KDD Data set into two different data sets SA and SF

SA is obtained by simply selecting all the normal data and a small proportion of abnormal data to gives an anomaly proportion of 1

SF is obtained as in 2 by simply picking up the data whose attribute loggedin is positive thus focusing on the

intrusion attack which gives a proportion of 03 of attack

http and smtp are two subsets of SF corresponding with third feature equal to 'http' resp to 'smtp'

General KDD structure

Samples total 4898431

Dimensionality 41

Features discrete int or continuous float

Targets str 'normal' or name of the anomaly type

SA structure

Samples total 976158

Dimensionality 41

Features discrete int or continuous float

Targets str 'normal' or name of the anomaly type

SF structure

Samples total 699691

Dimensionality 4

Features discrete int or continuous float

Targets str 'normal' or name of the anomaly type

642 Chapter 3 User Guide

scikitlearn user guide Release 0213

http structure

Samples total 619052

Dimensionality 3

Features discrete int or continuous float

Targets str 'normal' or name of the anomaly type

smtp structure

Samples total 95373

Dimensionality 3

Features discrete int or continuous float

Targets str 'normal' or name of the anomaly type

sklearn.datasets.fetch_kddcup99 will load the kddcup99 dataset it returns a dictionarylike object with the feature matrix in the data member and the target values in target The dataset will be downloaded from the web if necessary

California Housing dataset

Data Set Characteristics

Number of Instances 20640

Number of Attributes 8 numeric predictive attributes and the target

Attribute Information

- MedInc median income in block
- HouseAge median house age in block
- AveRooms average number of rooms
- AveBedrms average number of bedrooms
- Population block population
- AveOccup average house occupancy
- Latitude house block latitude
- Longitude house block longitude

Missing Attribute Values None

This dataset was obtained from the StatLib repository <http://lib.stat.cmu.edu/datasets>

The target variable is the median house value for California districts

This dataset was derived from the 1990 US census using one row per census block group A block group is the smallest geographical unit for which the US Census Bureau publishes sample data a block group typically has a population of 600 to 3000 people

It can be downloaded using the `sklearn.datasets.fetch_california_housing` function

References

36 Dataset loading utilities 643

scikitlearn user guide Release 0213

• Pace R Kelley and Ronald Barry Sparse Spatial Autoregressions Statistics and Probability Letters 33
1997 291297

364 Generated datasets

In addition scikitlearn includes various random sample generators that can be used to build artificial datasets of controlled size and complexity

Generators for classification and clustering

These generators produce a matrix of features and corresponding discrete targets

Single label

Bothmakeblobs andmakeclassification create multiclass datasets by allocating each class one or more normallydistributed clusters of points makeblobs provides greater control regarding the centers and standard deviations of each cluster and is used to demonstrate clustering makeclassification specialises in introducing noise by way of correlated redundant and uninformative features multiple Gaussian clusters per class and linear transformations of the feature space

makegaussianquantiles divides a single Gaussian cluster into nearequalsize classes separated

by concentric hyperspheres makehastie102 generates a similar binary 10dimensional problem

makecircles andmakemoons gener

ate 2d binary classification datasets that are challenging to certain algorithms eg centroidbased clustering or linear classification including optional Gaussian noise They are useful for visualisation makecircles produces Gaussian data with a spherical decision boundary for binary classification while makemoons produces two interleaving half circles

644 Chapter 3 User Guide

scikitlearn user guide Release 0213

Multilabel

makemultilabelclassification generates random samples with multiple labels reflecting a bag of words drawn from a mixture of topics The number of topics for each document is drawn from a Poisson distribution and the topics themselves are drawn from a fixed random distribution Similarly the number of words is drawn from Poisson with words drawn from a multinomial where each topic defines a probability distribution over words Simplifications with respect to true bagofwords mixtures include

- Pertopic word distributions are independently drawn where in reality all would be affected by a sparse base distribution and would be correlated
- For a document generated from multiple topics all topics are weighted equally in generating its bag of words
- Documents without labels words at random rather than from a base distribution

Biclustering

makebicclusters shape nclusters noise Generate an array with constant block diagonal structure for biclustering

makecheckerboard shape nclusters Generate an array with block checkerboard structure for bi clustering

Generators for regression

makeregression produces regression targets as an optionallysparse random linear combination of random fea tures with noise Its informative features may be uncorrelated or low rank few features account for most of the variance

Other regression generators generate functions deterministically from randomized features

makesparseuncorrelated produces a target as a linear combination of four features with fixed coef ficients Others encode explicitly nonlinear relations makefriedman1 is related by polynomial and sine transforms makefriedman2 includes feature multiplication and reciprocation and makefriedman3 is similar with an arctan transformation on the target

Generators for manifold learning

makescurve nsamples noise randomstate Generate an S curve dataset

makeswissroll nsamples noise randomstate Generate a swiss roll dataset

36 Dataset loading utilities 645

scikitlearn user guide Release 0.21.3

Generators for decomposition

`make_lowrank_matrix(nsamples, ...)` Generate a mostly low rank matrix with bell-shaped singular values

`make_sparse_coded_signal(nsamples, ...)` Generate a signal as a sparse combination of dictionary elements

`make_sparse_pdmatrix(ndim, random_state=...)` Generate a random symmetric positive definite matrix

`make_sparse_pdmatrix(dim, alpha=...)` Generate a sparse symmetric definite positive matrix

365 Loading other datasets

Sample images

Scikit-learn also embeds a couple of sample JPEG images published under Creative Commons license by their authors

Those images can be useful to test algorithms and pipeline on 2D data

`load_sample_images()` Load sample images for image manipulation

`load_sample_image(image_name)` Load the numpy array of a single sample image

Warning: The default coding of images is based on the `uint8` dtype to spare memory. Often machine learning

algorithms work best if the input is converted to a floating point representation first. Also if you plan to use

`matplotlib.pyplot.imshow()` don't forget to scale to the range `[0, 1]` as done in the following example

Examples

• Color Quantization using KMeans

Datasets in `svmlight` / `libsvm` format

scikit-learn includes utility functions for loading datasets in the `svmlight` / `libsvm` format. In this format each line

takes the form `label feature_id feature_value feature_id feature_value`

This format is especially suitable for sparse datasets. In this module `scipy.sparse` CSR (Compressed Sparse Row) matrices are used for `X` and

numpy arrays are used for `y`

You may load a dataset like as follows

```
from sklearn.datasets import load_svmlight_file
```

```
X_train, y_train = load_svmlight_file(path_to_train_dataset.txt)
```

```
scikitlearn user guide Release 0213
You may also load two or more datasets at once
Xtrain ytrain Xtest ytest loadsvmlightfiles
pathtotraindataset.txt pathtotestdataset.txt
```

In this case Xtrain and Xtest are guaranteed to have the same number of features Another way to achieve the same result is to fix the number of features

```
Xtest ytest loadsvmlightfile
pathtotestdataset.txt nfeaturesXtrain.shape[1]
```

```
Related links
Public datasets in svmlight libsvm format https://www.cs.tu-dortmund.de/~tjc/libsvmtools/datasets
Faster API compatible implementation https://github.com/blondel/svmlight_loader
Downloading datasets from the openml.org repository
openml.org is a public repository for machine learning data and experiments that allows everybody to upload open
datasets
The sklearn.datasets package is able to download datasets from the repository using the function sklearn
datasets.fetch_openml
For example to download a dataset of gene expressions in mice brains
from sklearn.datasets import fetch_openml
mice = fetch_openml(name='miceprotein', version=4)
To fully specify a dataset you need to provide a name and a version though the version is optional see Dataset
Versions below The dataset contains a total of 1080 examples belonging to 8 different classes
mice_data.shape
(1080, 77)
mice_target.shape
(1080,)
np.unique(mice_target)
array([0, 1, 2, 3, 4, 5, 6, 7])
dtype: object
You can get more information on the dataset by looking at the DESCR and details attributes
print(mice.DESCR)
Author: Clara Higuera, Katherine J. Gardiner, Krzysztof J. Cios
Source: UCI https://archive.ics.uci.edu/ml/datasets/MiceProteinExpression
↪ 2015
Please cite: Higuera C, Gardiner KJ, Cios KJ. 2015 Self-Organizing
Feature Maps Identify Proteins Critical to Learning in a Mouse Model of Down
Syndrome. PLoS ONE 10(6):e0129126
mice_details
id: 40966, name: MiceProtein, version: 4, format: ARFF
upload_date: 2017-11-08, T160015, licence: Public
36 Dataset loading utilities 647
```

scikitlearn user guide Release 0213
url <https://www.openml.org/data/v1/download/17928620/MiceProtein.arff>
fileid 17928620 defaulttargetattribute class
rowidattribute MouseID
ignoreattribute Genotype Treatment Behavior
tag OpenMLCC18 study135 study98 study99
visibility public status active
md5checksum 3c479a6885bfa0438971388283a1ce32
TheDESCR contains a freetext description of the data while details contains a dictionary of metadata stored by openml like the dataset id For more details see the OpenML documentation The dataid of the mice protein dataset is 40966 and you can use this or the name to get more information on the dataset on the openml website
miceurl
<https://www.openml.org/d/40966>
Thedataid also uniquely identifies a dataset from OpenML
mice fetchopenmldataid40966
micedetails
id 4550 name MiceProtein version 1 format ARFF
creator
uploaddate 20160217T143249 licence Public url
<https://www.openml.org/data/v1/download/1804243/MiceProtein.ARFF> fileid
1804243 defaulttargetattribute class citation Higuera C
Gardiner KJ Cios KJ 2015 SelfOrganizing Feature Maps Identify Proteins
Critical to Learning in a Mouse Model of Down Syndrome PLoS ONE 106
e0129126 Web Link journalpone0129126 tag OpenML100 study14
study34 visibility public status active md5checksum
3c479a6885bfa0438971388283a1ce32
Dataset Versions
A dataset is uniquely specified by its dataid but not necessarily by its name Several different “versions” of a dataset with the same name can exist which can contain entirely different datasets If a particular version of a dataset has been found to contain significant issues it might be deactivated Using a name to specify a dataset will yield the earliest version of a dataset that is still active That means that fetchopenmlnamemiceprotein can yield different results at different times if earlier versions become inactive You can see that the dataset with dataid 40966 that we fetched above is the version 1 of the “miceprotein” dataset
micedetailsversion
1
In fact this dataset only has one version The iris dataset on the other hand has multiple versions
iris fetchopenmlnameiris
irisdetailsversion
1
irisdetailsid
61
iris61 fetchopenmldataid61
iris61detailsversion
1
iris61detailsid
61
648 Chapter 3 User Guide

scikitlearn user guide Release 0213

iris969 fetchopenmldataid969

iris969detailsversion

3

iris969detailsid

969

Specifying the dataset by the name “iris” yields the lowest version version 1 with the dataid 61 To make sure you always get this exact dataset it is safest to specify it by the dataset dataid The other dataset with dataid 969 is version 3 version 2 has become inactive and contains a binarized version of the data

npuniqueiris969target

arrayN P dtypeobject

You can also specify both the name and the version which also uniquely identifies the dataset

irisversion3 fetchopenmlnameiris version3

irisversion3detailsversion

3

irisversion3detailsid

969

References

• Vanschoren van Rijn Bischl and Torgo “OpenML networked science in machine learning” ACM SIGKDD

Explorations Newsletter 152 4960 2014

Loading from external datasets

scikitlearn works on any numeric data stored as numpy arrays or scipy sparse matrices Other types that are convertible to numeric arrays such as pandas DataFrame are also acceptable

Here are some recommended ways to load standard columnar data into a format usable by scikitlearn

• pandasio provides tools to read data from common formats including CSV Excel JSON and SQL DataFrames may also be constructed from lists of tuples or dicts Pandas handles heterogeneous data smoothly and provides tools for manipulation and conversion into a numeric array suitable for scikitlearn

• scipyio specializes in binary formats often used in scientific computing context such as mat and arff

• numpyroutinesio for standard loading of columnar data into numpy arrays

• scikitlearn’s datasetsloadsvmlightfile for the svmlight or libSVM sparse format

• scikitlearn’s datasetsloadfiles for directories of text files where the name of each directory is the name of each category and each file inside of each directory corresponds to one sample from that category For some miscellaneous data such as images videos and audio you may wish to refer to

• skimageio or Imageio for loading images and videos into numpy arrays

• scipyiowavfileread for reading WAV files into a numpy array

Categorical or nominal features stored as strings common in pandas DataFrames will need converting to numerical features using sklearnpreprocessingOneHotEncoder or sklearnpreprocessing

OrdinalEncoder or similar See Preprocessing data

36 Dataset loading utilities 649

scikitlearn user guide Release 0213

Note if you manage your own numerical data it is recommended to use an optimized file format such as HDF5 to reduce data load times Various libraries such as H5Py PyTables and pandas provides a Python interface for reading and writing data in that format

37 Computing with scikitlearn

371 Strategies to scale computationally bigger data

For some applications the amount of examples features or both and/or the speed at which they need to be processed are challenging for traditional approaches In these cases scikitlearn has a number of options you can consider to make your system scale

Scaling with instances using outofcore learning

Outofcore or “external memory” learning is a technique used to learn from data that cannot fit in a computer’s main memory RAM

Here is a sketch of a system designed to achieve this goal

1 a way to stream instances

2 a way to extract features from instances

3 an incremental algorithm

Streaming instances

Basically 1 may be a reader that yields instances from files on a hard drive a database from a network stream etc

However details on how to achieve this are beyond the scope of this documentation

Extracting features

2 could be any relevant way to extract features among the different feature extraction methods supported by scikit learn However when working with data that needs vectorization and where the set of features or values is not known in advance one should take explicit care A good example is text classification where unknown terms are likely to be found during training It is possible to use a stateful vectorizer if making multiple passes over the data is reasonable from an application point of view Otherwise one can turn up the difficulty by using a stateless feature extractor Currently the preferred way to do this is to use the so-called hashing trick as implemented by sklearn featureextractionFeatureHasher for datasets with categorical variables represented as list of Python dicts or sklearnfeatureextractiontextHashingVectorizer for text documents

Incremental learning

Finally for 3 we have a number of options inside scikitlearn Although not all algorithms can learn incrementally ie without seeing all the instances at once all estimators implementing the partialfit API (Application Programming Interface) are candidates Actually the ability to learn incrementally from a minibatch of instances sometimes called “online learning” is key to outofcore learning as it guarantees that at any given time there will be only a small amount of instances in the

650 Chapter 3 User Guide

scikitlearn user guide Release 0213

main memory Choosing a good size for the minibatch that balances relevancy and memory footprint could involve some tuning1

Here is a list of incremental estimators for different tasks

- Classification
 - sklearnnaivebayesMultinomialNB
 - sklearnnaivebayesBernoulliNB
 - sklearnlinearmodelPerceptron
 - sklearnlinearmodelSGDClassifier
 - sklearnlinearmodelPassiveAggressiveClassifier
 - sklearnneuralnetworkMLPClassifier
- Regression
 - sklearnlinearmodelSGDRegressor
 - sklearnlinearmodelPassiveAggressiveRegressor
 - sklearnneuralnetworkMLPRegressor
- Clustering
 - sklearnclusterMiniBatchKMeans
 - sklearnclusterBirch
- Decomposition feature Extraction
 - sklearndecompositionMiniBatchDictionaryLearning
 - sklearndecompositionIncrementalPCA
 - sklearndecompositionLatentDirichletAllocation
- Preprocessing
 - sklearnpreprocessingStandardScaler
 - sklearnpreprocessingMinMaxScaler
 - sklearnpreprocessingMaxAbsScaler

For classification a somewhat important thing to note is that although a stateless feature extraction routine may be able to cope with newunseen attributes the incremental learner itself may be unable to cope with newunseen targets classes In this case you have to pass all the possible classes to the first partialfit call using the classes parameter

Another aspect to consider when choosing a proper algorithm is that not all of them put the same importance on each example over time Namely the Perceptron is still sensitive to badly labeled examples even after many examples whereas the SGDandPassiveAggressive families are more robust to this kind of artifacts Conversely the latter also tend to give less importance to remarkably different yet properly labeled examples when they come late in the stream as their learning rate decreases over time

1Depending on the algorithm the minibatch size can influence results or not SGD PassiveAggressive and discrete NaiveBayes are truly online and are not affected by batch size Conversely MiniBatchKMeans convergence rate is affected by the batch size Also its memory footprint can vary dramatically with batch size

37 Computing with scikitlearn 651

scikitlearn user guide Release 0213

Examples

Finally we have a fullfledged example of Outofcore classification of text documents. It is aimed at providing a starting point for people wanting to build outofcore learning systems and demonstrates most of the notions discussed above.

Furthermore it also shows the evolution of the performance of different algorithms with the number of processed examples.

Now looking at the computation time of the different parts we see that the vectorization is much more expensive than learning itself. From the different algorithms MultinomialNB is the most expensive but its overhead can be mitigated by increasing the size of the minibatches. Exercise: change minibatchsize to 100 and 10000 in the program and compare.

Notes

372 Computational Performance

For some applications the performance mainly latency and throughput at prediction time of estimators is crucial It may also be of interest to consider the training throughput but this is often less important in a production setup where it often takes place offline

We will review here the orders of magnitude you can expect from a number of scikitlearn estimators in different contexts and provide some tips and tricks for overcoming performance bottlenecks

Prediction latency is measured as the elapsed time necessary to make a prediction eg in microseconds Latency is often viewed as a distribution and operations engineers often focus on the latency at a given percentile of this distribution eg the 90 percentile

Prediction throughput is defined as the number of predictions the software can deliver in a given amount of time eg in predictions per second

An important aspect of performance optimization is also that it can hurt prediction accuracy Indeed simpler models eg linear instead of nonlinear or with fewer parameters often run faster but are not always able to take into account the same exact properties of the data as more complex ones

Prediction Latency

One of the most straightforward concerns one may have when usingchoosing a machine learning toolkit is the latency at which predictions can be made in a production environment

The main factors that influence the prediction latency are

- 1 Number of features

scikitlearn user guide Release 0213

2 Input data representation and sparsity

3 Model complexity

4 Feature extraction

A last major parameter is also the possibility to do predictions in bulk or oneatotime mode

Bulk versus Atomic mode

In general doing predictions in bulk many instances at the same time is more efficient for a number of reasons
branching predictability CPU cache linear algebra libraries optimizations etc Here we see on a setting with few
features that independently of estimator choice the bulk mode is always faster and for some of them by 1 to 2 orders
of magnitude

654 Chapter 3 User Guide

scikitlearn user guide Release 0213

To benchmark different estimators for your case you can simply change the nfeatures parameter in this example
Prediction Latency This should give you an estimate of the order of magnitude of the prediction latency

Configuring Scikitlearn for reduced validation overhead

Scikitlearn does some validation on data that increases the overhead per call to predict and similar functions
In particular checking that features are finite not NaN or infinite involves a full pass over the data If you en
sure that your data is acceptable you may suppress checking for finiteness by setting the environment variable
SKLEARNASSUMEFINITE to a nonempty string before importing scikitlearn or configure it in Python with
sklearnsetconfig For more control than these global settings a configcontext allows you to set this
configuration within a specified context

```
import sklearn
with sklearnconfigcontextassumefinite True
pass do learningprediction here with reduced validation
```

Note that this will affect all uses of sklearnutilsassertallfinite within the context

Influence of the Number of Features

Obviously when the number of features increases so does the memory consumption of each example Indeed for a
matrix of ninstances with nfeatures the space complexity is in $O(n^2)$ From a computing perspective it also
means that the number of basic operations eg multiplications for vectormatrix products in linear models increases
too Here is a graph of the evolution of the prediction latency with the number of features

37 Computing with scikitlearn 655

scikitlearn user guide Release 0213

Overall you can expect the prediction time to increase at least linearly with the number of features nonlinear cases can happen depending on the global memory footprint and estimator

Influence of the Input Data Representation

Scipy provides sparse matrix data structures which are optimized for storing sparse data The main feature of sparse formats is that you don't store zeros so if your data is sparse then you use much less memory A nonzero value in a sparse CSR (Compressed Sparse Row) or CSC representation will only take on average one 32bit integer position the 64 bit floating point value an additional 32bit per row or column in the matrix Using sparse input on a dense or sparse linear model can speedup prediction by quite a bit as only the non zero valued features impact the dot product and thus the model predictions Hence if you have 100 non zeros in 1e6 dimensional space you only need 100 multiply and add operation instead of 1e6

Calculation over a dense representation however may leverage highly optimised vector operations and multithreading in BLAS and tends to result in fewer CPU cache misses So the sparsity should typically be quite high 10 nonzeros max to be checked depending on the hardware for the sparse input representation to be faster than the dense input representation on a machine with many CPUs and an optimized BLAS implementation

Here is sample code to test the sparsity of your input

```
def sparsity_ratio(X):
    return 10 * np.count_nonzero(X) / float(X.shape[0] * X.shape[1])
print(input_sparsity_ratio(sparsity_ratio(X)))
```

As a rule of thumb you can consider that if the sparsity ratio is greater than 90 you can probably benefit from sparse formats Check Scipy's sparse matrix formats documentation for more information on how to build or convert your data to sparse matrix formats Most of the time the CSR (Compressed Sparse Row) and CSC formats work best

scikitlearn user guide Release 0.21.3

Influence of the Model Complexity

Generally speaking when model complexity increases predictive power and latency are supposed to increase. Increasing predictive power is usually interesting but for many applications we would better not increase prediction latency too much. We will now review this idea for different families of supervised models.

For sklearn.linear_model, e.g. Lasso, ElasticNet, SGDClassifier, Regressor, Ridge, RidgeClassifier, PassiveAggressiveClassifier, Regressor, LinearSVC, LogisticRegression, the decision function that is applied at prediction time is the same as a dot product, so latency should be equivalent.

Here is an example using sklearn.linear_model.StochasticGradientDescentClassifier with the elasticnet penalty. The regularization strength is globally controlled by the alpha parameter. With a sufficiently high alpha, one can then increase the l1_ratio parameter of elasticnet to enforce various levels of sparsity in the model coefficients. Higher sparsity here is interpreted as less model complexity as we need fewer coefficients to describe it fully. Of course sparsity influences in turn the prediction time as the sparse dot product takes time roughly proportional to the number of nonzero coefficients.

For the sklearn.svm family of algorithms with a nonlinear kernel, the latency is tied to the number of support vectors: the fewer, the faster. Latency and throughput should asymptotically grow linearly with the number of support vectors in a SVC (Support Vector Classification) or SVR (Support Vector Regression) model. The kernel will also influence the latency as it is a projection of the input vector once per support vector. In the following graph, the nu parameter of sklearn.svm.SVC was used to influence the number of support vectors.

3.7 Computing with scikitlearn 657

scikitlearn user guide Release 0213

For sklearnensemble of trees eg RandomForest GBT ExtraTrees etc the number of trees and their depth play the most important role Latency and throughput should scale linearly with the number of trees In this case we used directly the nestimators parameter of sklearnensemblegradientboosting

GradientBoostingRegressor

In any case be warned that decreasing model complexity can hurt accuracy as mentioned above For instance a non linearly separable problem can be handled with a speedy linear model but prediction power will very likely suffer in the process

658 Chapter 3 User Guide

scikitlearn user guide Release 0213

Feature Extraction Latency

Most scikitlearn models are usually pretty fast as they are implemented either with compiled Cython extensions or optimized computing libraries On the other hand in many real world applications the feature extraction process ie turning raw data like database rows or network packets into numpy arrays governs the overall prediction time For example on the Reuters text classification task the whole preparation reading and parsing SGML files tokenizing the text and hashing it into a common vector space is taking 100 to 500 times more time than the actual prediction code depending on the chosen model

In many cases it is thus recommended to carefully time and profile your feature extraction code as it may be a good place to start optimizing when your overall latency is too slow for your application

Prediction Throughput

Another important metric to care about when sizing production systems is the throughput ie the number of predictions you can make in a given amount of time Here is a benchmark from the Prediction Latency example that measures this quantity for a number of estimators on synthetic data

37 Computing with scikitlearn 659

scikitlearn user guide Release 0213

These throughputs are achieved on a single process An obvious way to increase the throughput of your application is to spawn additional instances usually processes in Python because of the GIL that share the same model One might also add machines to spread the load A detailed explanation on how to achieve this is beyond the scope of this documentation though

Tips and Tricks

Linear algebra libraries

As scikitlearn relies heavily on NumpyScipy and linear algebra in general it makes sense to take explicit care of the versions of these libraries Basically you ought to make sure that Numpy is built using an optimized BLAS LAPACK library

Not all models benefit from optimized BLAS and Lapack implementations For instance models based on randomized decision trees typically do not rely on BLAS calls in their inner loops nor do kernel SVMs SVCSVRNuSVC NuSVR On the other hand a linear model implemented with a BLAS DGEMM call via numpydot will typically benefit hugely from a tuned BLAS implementation and lead to orders of magnitude speedup over a nonoptimized BLAS

You can display the BLAS LAPACK implementation used by your NumPy SciPy scikitlearn install with the following commands

```
from numpydistutilssysteminfo import getinfo
printgetinfoblasopt
printgetinfo lapackopt
```

Optimized BLAS LAPACK implementations include

- Atlas need hardware specific tuning by rebuilding on the target machine

scikitlearn user guide Release 0213

- OpenBLAS
- MKL
- Apple Accelerate and vecLib frameworks OSX only

More information can be found on the Scipy install page and in this blog post from Daniel Nouri which has some nice step by step install instructions for Debian Ubuntu

Limiting Working Memory

Some calculations when implemented using standard numpy vectorized operations involve using a large amount of temporary memory This may potentially exhaust system memory Where computations can be performed in fixed memory chunks we attempt to do so and allow the user to hint at the maximum size of this working memory defaulting to 1GB using sklearn.set_config or config_context The following suggests to limit temporary working memory to 128 MiB

```
import sklearn
with sklearn.config_context(working_memory=128):
    pass # do chunked work here
```

An example of a chunked operation adhering to this setting is metric.pairwise_distances_chunked which facilitates computing rowwise reductions of a pairwise distance matrix

Model Compression

Model compression in scikitlearn only concerns linear models for the moment In this context it means that we want to control the model sparsity ie the number of nonzero coordinates in the model vectors It is generally a good idea to combine model sparsity with sparse input data representation

Here is sample code that illustrates the use of the sparsify method

```
clf = SGDRegressor(penalty='elasticnet', l1_ratio=0.25)
clf.fit(X_train, y_train)
clf.predict(X_test)
```

In this example we prefer the elasticnet penalty as it is often a good compromise between model compactness and prediction power One can also further tune the l1_ratio parameter in combination with the regularization strength alpha to control this tradeoff

A typical benchmark on synthetic data yields a 30% decrease in latency when both the model and input are sparse with 0.000024 and 0.027400 nonzero coefficients ratio respectively Your mileage may vary depending on the sparsity and size of your data and model Furthermore sparsifying can be very useful to reduce the memory usage of predictive models deployed on production servers

Model Reshaping

Model reshaping consists in selecting only a portion of the available features to fit a model In other words if a model discards features during the learning phase we can then strip those from the input This has several benefits Firstly it reduces memory and therefore time overhead of the model itself It also allows to discard explicit feature selection components in a pipeline once we know which features to keep from a previous run Finally it can help reduce processing time and IO usage upstream in the data access and feature extraction layers by not collecting and building features that are discarded by the model For instance if the raw data come from a database it can make it possible to write simpler and faster queries or reduce IO usage by making the queries return lighter records At the 37 Computing with scikitlearn 661

scikitlearn user guide Release 0213

moment reshaping needs to be performed manually in scikitlearn In the case of sparse input particularly in CSR (Compressed Sparse Row) format it is generally sufficient to not generate the relevant features leaving their columns empty

Links

- scikitlearn developer performance documentation
- Scipy sparse matrix formats documentation

373 Parallelism resource management and configuration

Parallel and distributed computing

Scikitlearn uses the joblib library to enable parallel computing inside its estimators See the joblib documentation for the switches to control parallel computing

Note that by default scikitlearn uses its embedded vendored version of joblib A configuration switch documented below controls this behavior

Configuration switches

Python runtime

sklearnsetconfig controls the following behaviors

assumefinite used to skip validation which enables faster computations but may lead to segmentation faults if the data contains NaNs

workingmemory the optimal size of temporary arrays used by some algoritms

Environment variables

These environment variables should be set before importing scikitlearn

SKLEARNSITEJOBLIB When this environment variable is set to a non zero value scikitlearn uses the site joblib rather than its vendored version Consequently joblib must be installed for scikitlearn to run Note that using the site joblib is at your own risks the versions of scikitlearn and joblib need to be compatible Currently joblib 011 is supported In addition dumps from joblibMemory might be incompatible and you might loose some caches and have to redownload some datasets

Deprecated since version 021 As of version 021 this parameter has no effect vendored joblib was removed and site joblib is always used

SKLEARNASSUMEFINITE Sets the default value for the assumefinite argument of sklearnsetconfig

SKLEARNWORKINGMEMORY Sets the default value for the Limiting Working Memory argument ofsklearnsetconfig

SKLEARNSEED Sets the seed of the global random generator when running the tests for reproducibility

SKLEARNSKIPNETWORKTESTS When this environment variable is set to a non zero value the tests that need network access are skipped

662 Chapter 3 User Guide

CHAPTER

FOUR

GLOSSARY OF COMMON TERMS AND API (Application Programming Interface) ELEMENTS

This glossary hopes to definitively represent the tacit and explicit conventions applied in Scikitlearn and its API (Application Programming Interface) providing a reference for users and contributors It aims to describe the concepts and either detail their corresponding API (Application Programming Interface) or link to other relevant parts of the documentation which do so By linking to glossary entries from the API Reference and User Guide we may minimize redundancy and inconsistency

We begin by listing general concepts and any that didn't fit elsewhere but more specific sets of related terms are listed below Class APIs and Estimator Types Target Types Methods Parameters Attributes Data and sample properties

41 General Concepts

1d

1d array Onedimensional array A NumPy array whose shape has length 1 A vector

2d

2d array Twodimensional array A NumPy array whose shape has length 2 Often represents a matrix

API (Application Programming Interface) Refers to both the specific interfaces for estimators implemented in Scikitlearn and the generalized conventions across types of estimators as described in this glossary and overviewed in the contributor documentation

The specific interfaces that constitute Scikitlearn's public API (Application Programming Interface) are largely documented in API (Application Programming Interface) ever we less formally consider anything as public API (Application Programming Interface) if none of the identifiers required to access it begins with an underscore

We generally try to maintain backwards compatibility for all objects in the public API (Application Programming Interface)

Private API (Application Programming Interface) including functions modules and methods beginning with an underscore are not assured to be stable

arraylike The most common data format for input to Scikitlearn estimators and functions arraylike is any type that supports the `__getitem__` and `__len__` methods

object for which `numpy.asarray` will produce an array of appropriate shape usually 1 or 2dimensional of appropriate dtype usually numeric

This includes

- a numpy array
- a list of numbers
- a list of lengthk lists of numbers for some fixed length k
- a pandasDataFrame with all columns numeric
- a numeric pandasSeries

It excludes

- a sparse matrix
- an iterator

scikitlearn user guide Release 0213

- a generator

Note that output from scikitlearn estimators and functions eg predictions should generally be arrays or sparse matrices or lists thereof as in multioutput treeDecisionTreeClassifier 'spredictproba An estimator where predict returns a list or a pandasSeries is not valid

attribute

attributes We mostly use attribute to refer to how model information is stored on an estimator during fitting Any public attribute stored on an estimator instance is required to begin with an alphabetic character and end in a single underscore if it is set in fitorpartialfit These are what is documented under an estimator's Attributes documentation The information stored in attributes is usually either sufficient statistics used for prediction or transformation transductive outputs such as labels orembedding or diagnostic data such as featureimportances Common attributes are listed below

A public attribute may have the same name as a constructor parameter with a appended This is used to store a validated or estimated version of the user's input For example decompositionPCA is constructed with anncomponents parameter From this together with other parameters and the data PCA (Principal Component Analysis) estimates the attributencomponents

Further private attributes used in predictiontransformationetc may also be set when fitting These begin with a single underscore and are not assured to be stable for public access

A public attribute on an estimator instance that does not end in an underscore should be the stored unmodified value of an init parameter of the same name Because of this equivalence these are documented under an estimator's Parameters documentation

backwards compatibility We generally try to maintain backwards compatibility ie interfaces and behaviors may be extended but not changed or removed from release to release but this comes with some exceptions

Public API (Application Programming Interface) only The behaviour of objects accessed through private identifiers those beginning may be changed arbitrarily between versions

As documented We will generally assume that the users have adhered to the documented parameter types and ranges If the documentation asks for a list and the user gives a tuple we do not assure consistent behavior from version to version

Deprecation Behaviors may change following a deprecation period usually two releases long Warnings are issued using Python's warnings module

Keyword arguments We may sometimes assume that all optional parameters other than X and y to fitand similar methods are passed as keyword arguments only and may be positionally reordered

Bug fixes and enhancements Bug fixes and - less often - enhancements may change the behavior of estimators including the predictions of an estimator trained on the same data and randomstate When this happens we attempt to note it clearly in the changelog

Serialization We make no assurances that pickling an estimator in one version will allow it to be unpickled to an equivalent model in the subsequent version For estimators in the sklearn package we issue a warning when this unpickling is attempted even if it may happen to work See Security maintainability limitations

utilsestimatorcheckscheckestimator We provide limited backwards compatibility assurances for the estimator checks we may add extra requirements on estimators tested with this function usually when these were informally assumed but not formally tested

Despite this informal contract with our users the software is provided as is as stated in the licence When a release inadvertently introduces changes that are not backwards compatible these are known as software regressions

callable A function class or an object which implements the call method anything that returns True when the argument of callable

scikitlearn user guide Release 0213

categorical feature A categorical or nominal feature is one that has a finite set of discrete values across the population of data These are commonly represented as columns of integers or strings Strings will be rejected by most scikitlearn estimators and integers will be treated as ordinal or countvalued For the use with most estimators categorical variables should be onehot encoded Notable exceptions include treebased models such as random forests and gradient boosting models that often work better and faster with integercoded categorical variables OrdinalEncoder helps encoding stringvalued categorical features as ordinal integers and OneHotEncoder can be used to onehot encode categorical features See also Encoding categorical features and the categoricalencoding package for tools related to encoding categorical features

clone

cloned To copy an estimator instance and create a new one with identical parameters but without any fitted attributes usingclone

Whenfit is called a metaestimator usually clones a wrapped estimator instance before fitting the cloned instance Exceptions for legacy reasons include Pipeline andFeatureUnion

common tests This refers to the tests run on almost every estimator class in Scikitlearn to check they comply with basic API (Application Programming Interface) conventions They are available for external use through utilsestimatorchecks checkestimator with most of the implementation in sklearnutilsestimatorcheckspy

Note Some exceptions to the common testing regime are currently hardcoded into the library but we hope to replace this by marking exceptional behaviours on the estimator using semantic estimator tags

deprecation We use deprecation to slowly violate our backwards compatibility assurances usually to to

- change the default value of a parameter or
- remove a parameter attribute method class etc

We will ordinarily issue a warning when a deprecated element is used although there may be limitations to this

For instance we will raise a warning when someone sets a parameter that has been deprecated but may not when they access that parameter's attribute on the estimator instance

See the Contributors' Guide

dimensionality May be used to refer to the number of features ie nfeatures or columns in a 2d feature matrix Dimensions are however also used to refer to the length of a NumPy array's shape distinguishing a 1d array from a 2d matrix

docstring The embedded documentation for a module class function etc usually in code as a string at the beginning of the object's definition and accessible as the object's doc attribute

We try to adhere to PEP257 and follow NumpyDoc conventions

double underscore

double underscore notation When specifying parameter names for nested estimators may be used to separate between parent and child in some contexts The most common use is when setting parameters through a meta estimator with setparams and hence in specifying a search grid in parameter search See parameter It is also used inpipelinePipelinefit for passing sample properties to thefit methods of estimators in the

pipeline

dtype

data type NumPy arrays assume a homogeneous data type throughout available in the dtype attribute of an array or sparse matrix We generally assume simple data types for scikitlearn data float or integer We may support object or string data types for arrays before encoding or vectorizing Our estimators do not work with struct arrays for instance

TODO Mention efficiency and precision issues casting policy

41 General Concepts 665

scikitlearn user guide Release 0213

duck typing We try to apply duck typing to determine how to handle some input values eg checking whether a given estimator is a classifier That is we avoid using isinstance where possible and rely on the presence or absence of attributes to determine an object’s behaviour Some nuance is required when following this approach

- For some estimators an attribute may only be available once it is fitted For instance we cannot a priori determine if predict_proba is available in a grid search where the grid includes alternating between a probabilistic and a nonprobabilistic predictor in the final step of the pipeline In the following we can only determine if clf is probabilistic after fitting it on some data

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import SGDClassifier
clf = GridSearchCV(SGDClassifier(
    param_grid=dict(loss=log, hinge=
```

This means that we can only check for ducktyped attributes after fitting and that we must be careful to make metaestimators only present attributes according to the state of the underlying estimator after fitting

- Checking if an attribute is present using hasattr is in general just as expensive as getting the attribute getattr or dot notation In some cases getting the attribute may indeed be expensive eg for some implementations of feature importances which may suggest this is an API (Application Programming Interface) design flaw So code which does hasattr followed by getattr should be avoided getattr within a tryexcept block is preferred

- For determining some aspects of an estimator’s expectations or support for some feature we use estimator tags instead of duck typing

early stopping This consists in stopping an iterative optimization method before the convergence of the training loss to avoid overfitting This is generally done by monitoring the generalization score on a validation set When available it is activated through the parameter early_stopping or by setting a positive n_iter_no_change estimator instance We sometimes use this terminology to distinguish an estimator class from a constructed instance For example in the following cls is an estimator class while est1 and est2 are instances

```
cls = RandomForestClassifier
est1 = cls
est2 = RandomForestClassifier
```

examples We try to give examples of basic usage for most functions and classes in the API (Application Programming Interface)

- as doctests in their docstrings ie within the sklearn library code itself
- as examples in the example gallery rendered using sphinx_gallery from scripts in the examples directory exemplifying key features or parameters of the estimator/function These should also be referenced from the User Guide

- sometimes in the User Guide built from doc alongside a technical description of the estimator evaluation metric

evaluation metrics Evaluation metrics give a measure of how well a model performs We may use this term specifically to refer to the functions in metrics disregarding metrics pairwise as distinct from the score method and the scoring API (Application Programming Interface) used in cross validation See Model evaluation quantifying the quality of predictions

These functions usually accept a ground truth or the raw data where the metric evaluates clustering without a ground truth and a prediction be it the output of predict ypred of predict_proba yproba or of an arbitrary score function including decisionfunction yscore Functions are usually named to end with score if a greater score indicates a better model and loss if a lesser score indicates a better model This diversity of interface motivates the scoring API (Application Programming Interface)

scikitlearn user guide Release 0213

Note that some estimators can calculate metrics that are not included in metrics and are estimatorspecific notably model likelihoods

estimator tags A proposed feature eg 8022 by which the capabilities of an estimator are described through a set of semantic tags This would enable some runtime behaviors based on estimator inspection but it also allows each estimator to be tested for appropriate invariances while being excepted from other common tests Some aspects of estimator tags are currently determined through the duck typing of methods like predictproba and through some special attributes on estimator objects

estimatortype This stringvalued attribute identifies an estimator as being a classifier regressor etc It is set by mixins such as baseClassifierMixin but needs to be more explicitly adopted on a meta estimator Its value should usually be checked by way of a helper such as baseisclassifier pairwise This boolean attribute indicates whether the data X passed tofit and similar methods consists of pairwise measures over samples rather than a feature representation for each sample It is usually True where an estimator has a metric oraffinity orkernel parameter with value 'precomputed' Its primary purpose is that when a metaestimator extracts a subsample of data intended for a pairwise estimator the data needs to be indexed on both axes while other data is indexed only on the first axis

feature

features

feature vector In the abstract a feature is a function in its mathematical sense mapping a sampled object to a numeric or categorical quantity "Feature" is also commonly used to refer to these quantities being the individual elements of a vector representing a sample In a data matrix features are represented as columns each column contains the result of applying a feature function to a set of samples

Elsewhere features are known as attributes predictors regressors or independent variables

Nearly all estimators in scikitlearn assume that features are numeric finite and not missing even when they have semantically distinct domains and distributions categorical ordinal countvalued realvalued interval See also categorical feature andmissing values

nfeatures indicates the number of features in a dataset

fitting Calling fitforfittransform fitpredict etc on an estimator

fitted The state of an estimator after fitting

There is no conventional procedure for checking if an estimator is fitted However an estimator that is not fitted

- should raise exceptionsNotFittedError when a prediction method predict transform etc is

called utilvalidationcheckisfitted is used internally for this purpose

- should not have any attributes beginning with an alphabetic character and ending with an underscore

Note that a descriptor for the attribute may still be present on the class but hasattr should return False

function We provide ad hoc function interfaces for many algorithms while estimator classes provide a more consistent interface

In particular Scikitlearn may provide a function interface that fits a model to some data and returns the learnt

model parameters as in linearmodelenetpath For transductive models this also returns the em

bedding or cluster labels as in manifoldspectralembedding orclusterdbscan Many prepro

cessing transformers also provide a function interface akin to calling fittransform as inpreprocessing

maxabsscale Users should be careful to avoid data leakage when making use of these fittransform

equivalent functions

We do not have a strict policy about when to or when not to provide function forms of estimators but maintainers should consider consistency with existing interfaces and whether providing a function would lead users astray

from best practices as regards data leakage etc

41 General Concepts 667

scikitlearn user guide Release 0213
gallery Seeexamples
hyperparameter
hyperparameter Seeparameter
impute
imputation Most machine learning algorithms require that their inputs have no missing values and will not work if this requirement is violated Algorithms that attempt to fill in or impute missing values are referred to as imputation algorithms
indexable Anarraylike sparse matrix pandas DataFrame or sequence usually a list
induction
inductive Inductive contrasted with transductive machine learning builds a model of some data that can then be applied to new instances Most estimators in Scikitlearn are inductive having predict and/or transform methods
joblib A Python library <https://joblib.readthedocs.io> used in Scikitlearn to facilitate simple parallelism and caching
Joblib is oriented towards efficiently working with numpy arrays such as through use of memory mapping See Parallel and distributed computing for more information
label indicator matrix
multilabel indicator matrix
multilabel indicator matrices The format used to represent multilabel data where each row of a 2d array or sparse matrix corresponds to a sample each column corresponds to a class and each element is 1 if the sample is labeled with the class and 0 if not
leakage
data leakage A problem in cross validation where generalization performance can be overestimated since knowledge of the test data was inadvertently included in training a model This is a risk for instance when applying a transformer to the entirety of a dataset rather than each training portion in a cross validation split
We aim to provide interfaces such as pipeline and model selection that shield the user from data leakage
memmapping
memory map
memory mapping A memory efficiency strategy that keeps data on disk rather than copying it into main memory
Memory maps can be created for arrays that can be read written or both using `numpy.memmap` When using joblib to parallelize operations in Scikitlearn it may automatically memmap large arrays to reduce memory duplication overhead in multiprocessing
missing values Most Scikitlearn estimators do not work with missing values When they do eg in `SimpleImputer` `NaN` is the preferred representation of missing values in float arrays If the array has integer dtype `NaN` cannot be represented For this reason we support specifying another missing values value when imputation or learning can be performed in integer space Unlabeled data is a special case of missing values in the target
`nfeatures` The number of features
`noutputs` The number of outputs in the target
`nsamples` The number of samples
`ntargets` Synonym for `noutputs`
narrative docs
668 Chapter 4 Glossary of Common Terms and API (Application Programming Interface) Elements

scikitlearn user guide Release 0213

narrative documentation An alias for User Guide ie documentation written in docmodules Unlike the API (Application Programming Interface) reference provided through docstrings the User Guide aims to

- group tools provided by Scikitlearn together thematically or in terms of usage
- motivate why someone would use each particular tool often through comparison
- provide both intuitive and technical descriptions of tools
- provide or link to examples of using key features of a tool

npA shorthand for Numpy due to the conventional import statement

import numpy as np

online learning Where a model is iteratively updated by receiving each batch of ground truth targets soon after making predictions on corresponding batch of data Intrinsically the model must be usable for prediction after each batch See partialfit

outofcore An efficiency strategy where not all the data is stored in main memory at once usually by performing learning on batches of data See partialfit

outputs Individual scalarcategorical variables per sample in the target For example in multilabel classification each possible label corresponds to a binary output Also called responses tasks ortargets See multiclass multioutput andcontinuous multioutput

pair A tuple of length two

parameter

parameters

param

params We mostly use parameter to refer to the aspects of an estimator that can be specified in its construction For examplemaxdepth andrandomstate are parameters of RandomForestClassifier Parameters

to an estimator’s constructor are stored unmodified as attributes on the estimator instance and conventionally start with an alphabetic character and end with an alphanumeric character Each estimator’s constructor parameters are described in the estimator’s docstring

We do not use parameters in the statistical sense where parameters are values that specify a model and can be estimated from data What we call parameters might be what statisticians call hyperparameters to the model aspects for configuring model structure that are often not directly learnt from data However our parameters are also used to prescribe modeling operations that do not affect the learnt model such as njobs for controlling parallelism

When talking about the parameters of a metaestimator we may also be including the parameters of the estimators wrapped by the metaestimator Ordinarily these nested parameters are denoted by using adouble underscore to separate between the estimatorasparameter and its parameter Thus clf

BaggingClassifierbaseestimatorDecisionTreeClassifiermaxdepth3 has

a deep parameter baseestimatormaxdepth with value 3 which is accessible with clf

baseestimatormaxdepth orclfgetparamsbaseestimatormaxdepth

The list of parameters and their current values can be retrieved from an estimator instance using its getparams method

Between construction and fitting parameters may be modified using setparams To enable this parameters are not ordinarily validated or altered when the estimator is constructed or when each parameter is set Parameter validation is performed when fitis called

Common parameters are listed below

pairwise metric

41 General Concepts 669

scikitlearn user guide Release 0213

pairwise metrics In its broad sense a pairwise metric defines a function for measuring similarity or dissimilarity between two samples with each ordinarily represented as a feature vector We particularly provide implementations of distance metrics as well as improper metrics like Cosine Distance through metrics pairwise_distances and of kernel functions a constrained class of similarity functions in metrics pairwise_kernels These can compute pairwise distance matrices that are symmetric and hence store data redundantly

See also precomputed and metric

Note that for most distance metrics we rely on implementations from scipy.spatial.distance but may reimplement for efficiency in our context The neighbors module also duplicates some metric implementations for integration with efficient binary tree search data structures

pdA shorthand for Pandas due to the conventional import statement

import pandas as pd

precomputed Where algorithms rely on pairwise metrics and can be computed from pairwise metrics alone we often allow the user to specify that the X provided is already in the pairwise dissimilarity space rather than in a feature space That is when passed to fit it is a square symmetric matrix with each vector indicating dissimilarity to every sample and when passed to prediction transformation methods each row corresponds to a testing sample and each column to a training sample

Use of precomputed X is usually indicated by setting a metric affinity or kernel parameter to the string 'precomputed' An estimator should mark itself as being pairwise if this is the case rectangular Data that can be represented as a matrix with samples on the first axis and a fixed finite set of features on the second is called rectangular

This term excludes samples with nonvectorial structure such as text an image of arbitrary size a time series of arbitrary length a set of vectors etc The purpose of a vectorizer is to produce rectangular forms of such data sample

samples We usually use this term as a noun to indicate a single feature vector Elsewhere a sample is called an instance data point or observation nsamples indicates the number of samples in a dataset being the number of rows in a data array X

sample property

sample properties A sample property is data for each sample eg an array of length nsamples passed to an estimator method or a similar function alongside but distinct from the features X and target y The most prominent example is sample_weight see others at Data and sample properties

As of version 019 we do not have a consistent approach to handling sample properties and their routing in metaestimators though a fit_params parameter is often used

scikitlearncontrib A venue for publishing Scikitlearncompatible libraries that are broadly authorized by the core developers and the contrib community but not maintained by the core developer team See https

scikitlearncontribgithubio

scikitlearn enhancement proposals

SLEP

SLEPs Changes to the API (Application Programming Interface) principles and changes to dependencies or supported versions happen via a SLEP and

follows the decisionmaking process outlined in Scikitlearn governance and decisionmaking For all votes a proposal must have been made public and discussed before the vote Such proposal must be a consolidated document in the form of a 'ScikitLearn Enhancement Proposal' SLEP rather than a long discussion on an issue A SLEP must be submitted as a pullrequest to enhancement proposals using the SLEP template

semisupervised

670 Chapter 4 Glossary of Common Terms and API (Application Programming Interface) Elements

scikitlearn user guide Release 0213

semisupervised learning

semisupervised Learning where the expected prediction label or ground truth is only available for some samples provided as training data when fitting the model We conventionally apply the label 1 to unlabeled samples in semisupervised classification

sparse matrix A representation of two dimensional numeric data that is more memory efficient than the corresponding dense numpy array where almost all elements are zero We use the scipy sparse framework which provides several underlying sparse data representations or formats Some formats are more efficient than others for particular tasks and when a particular format provides especial benefit we try to document this fact in Scikit learn parameter descriptions

Some sparse matrix formats notably CSR (Compressed Sparse Row) CSC COO and LIL distinguish between implicit and explicit zeros Explicit zeros are stored ie they consume memory in a data array in the data structure while implicit zeros correspond to every element not otherwise defined in explicit storage

Two semantics for sparse matrices are used in Scikit learn

matrix semantics The sparse matrix is interpreted as an array with implicit and explicit zeros being interpreted as the number 0 This is the interpretation most often adopted eg when sparse matrices are used for feature matrices or multilabel indicator matrices

graph semantics As with scipy sparse csrgraph explicit zeros are interpreted as the number 0 but implicit zeros indicate a masked or absent value such as the absence of an edge between two vertices of a graph where an explicit value indicates an edge's weight This interpretation is adopted to represent connectivity in clustering in representations of nearest neighborhoods eg neighbors kneighborsgraph and for precomputed distance representation where only distances in the neighborhood of each point are required

When working with sparse matrices we assume that it is sparse for a good reason and avoid writing code that densifies a user provided sparse matrix instead maintaining sparsity or raising an error if not possible ie if an estimator does not cannot support sparse matrices

supervised

supervised learning Learning where the expected prediction label or ground truth is available for each sample when fitting the model provided as y This is the approach taken in a classifier or regressor among other estimators

target

targets The dependent variable in supervised and semisupervised learning passed as y to an estimator's fit method Also known as dependent variable outcome variable response variable ground truth or label Scikit learn works with targets that have minimal structure a class from a finite set a finite real valued number multiple classes or multiple numbers See Target Types

transduction

transductive A transductive contrasted with inductive machine learning method is designed to model a specific dataset but not to apply that model to unseen data Examples include manifold TSNE cluster Agglomerative Clustering and neighbors Local Outlier Factor

unlabeled

unlabeled data Samples with an unknown ground truth when fitting equivalently missing values in the target See also semisupervised and unsupervised learning

unsupervised

unsupervised learning Learning where the expected prediction label or ground truth is not available for each sample when fitting the model as in clusterers and outlier detectors Unsupervised estimators ignore any y passed to fit

42 Class APIs and Estimator Types

classifier

classifiers A supervised or semi-supervised predictor with a finite set of discrete possible output values

A classifier supports modeling some of binary multiclass multilabel or multiclass multioutput targets Within scikitlearn all classifiers support multiclass classification defaulting to using a one-vs-rest strategy over the binary classification problem

Classifiers must store a classes attribute after fitting and usually inherit from baseClassifierMixin

which sets their estimator_type attribute

A classifier can be distinguished from other estimators with is_classifier

A classifier must implement

- fit
- predict
- score

It may also be appropriate to implement decision_function predict_proba and predict_log_proba

clusterer

clusterers An unsupervised predictor with a finite set of discrete output values

A clusterer usually stores labels after fitting and must do so if it is transductive

A clusterer must implement

- fit
- fit_predict if transductive
- predict if inductive

density estimator TODO

estimator

estimators An object which manages the estimation and decoding of a model The model is estimated as a deterministic function of

- parameters provided in object construction or with set_params
- the global numpy.random random state if the estimator's random_state parameter is set to None and
- any data or sample properties passed to the most recent call to fit fit_transform or fit_predict or data

similarly passed in a sequence of calls to partial_fit

The estimated model is stored in public and private attributes on the estimator instance facilitating decoding through prediction and transformation methods

Estimators must provide a fit_method and should provide set_params and get_params although these are usually provided by inheritance from baseBaseEstimator

The core functionality of some estimators may also be available as a function

feature extractor

feature extractors A transformer which takes input where each sample is not represented as an arraylike object of fixed length and produces an arraylike object of features for each sample and thus a 2-dimensional arraylike for a set of samples In other words it lossily maps a nonrectangular data representation into rectangular data

scikitlearn user guide Release 0213

Feature extractors must implement at least

- fit
- transform
- getfeaturenames

metaestimator

metaestimators

metaestimator

metaestimators Anestimator which takes another estimator as a parameter Examples include pipeline

Pipeline modelselectionGridSearchCV featureselectionSelectFromModel and

ensembleBaggingClassifier

In a metaestimator’s fitmethod any contained estimators should be cloned before they are fit although FIXME

Pipeline and FeatureUnion do not do this currently An exception to this is that an estimator may explic

itly document that it accepts a prefitted estimator eg using prefitTrue infeatureselection

SelectFromModel One known issue with this is that the prefitted estimator will lose its model if the

metaestimator is cloned A metaestimator should have fit called before prediction even if all contained

estimators are prefitted

In cases where a metaestimator’s primary behaviors eg predict ortransform implementation are functions

of predictiontransformation methods of the provided base estimator or multiple base estimators a meta

estimator should provide at least the standard methods provided by the base estimator It may not be possible

to identify which methods are provided by the underlying estimator until the metaestimator has been fitted

see also duck typing for which utilsmetaestimatorsifdelegatehasmethod may help It

should also provide or modify the estimator tags andclasses attribute provided by the base estimator

Metaestimators should be careful to validate data as minimally as possible before passing it to an underlying

estimator This saves computation time and may for instance allow the underlying estimator to easily work

with data that is not rectangular

outlier detector

outlier detectors Anunsupervised binary predictor which models the distinction between core and outlying samples

Outlier detectors must implement

- fit
- fitpredict iftransductive
- predict ifinductive

Inductive outlier detectors may also implement decisionfunction to give a normalized inlier score where outliers

have score below 0 scoresamples may provide an unnormalized score per sample

predictor

predictors Anestimator supporting predict andor fitpredict This encompasses classifier regressor outlier detec

torandclusterer

In statistics “predictors” refers to features

regressor

regressors Asupervised orsemisupervised predictor with continuous output values

Regressors usually inherit from baseRegressorMixin which sets their estimatortype attribute

A regressor can be distinguished from other estimators with isregressor

42 Class APIs and Estimator Types 673

scikitlearn user guide Release 0213

A regressor must implement

- fit
- predict
- score

transformer

transformers An estimator supporting transform and/or fit_transform A purely transductive transformer such as manifoldTSNE may not implement transform

vectorizer

vectorizers See feature extractor

There are further APIs specifically related to a small family of estimators such as

crossvalidation splitter

CV (Cross-Validation) splitter

crossvalidation generator A non-estimator family of classes used to split a dataset into a sequence of train and test portions see Crossvalidation evaluating estimator performance by providing split and get_n_splits methods

Note that unlike estimators these do not have fit methods and do not provide set_params or get_params

Parameter validation may be performed in init

crossvalidation estimator An estimator that has builtin crossvalidation capabilities to automatically select the best

hyperparameters see the User Guide Some example of crossvalidation estimators are ElasticNetCV and

LogisticRegressionCV Crossvalidation estimators are named EstimatorCV and tend to be roughly

equivalent to GridSearchCVEstimator The advantage of using a crossvalidation estimator

over the canonical Estimator class along with grid search is that they can take advantage of warmstarting

by reusing precomputed results in the previous steps of the crossvalidation process This generally leads to

speed improvements An exception is the RidgeCV class which can instead perform efficient LeaveOneOut

CV (Cross-Validation)

scorer A non-estimator callable object which evaluates an estimator on given test data returning a number Unlike

evaluation metrics a greater returned number must correspond with a better score See The scoring parameter

defining model evaluation rules

Further examples

- neighborsDistanceMetric
- gaussianprocesskernelsKernel
- treeCriterion

43 Target Types

binary A classification problem consisting of two classes A binary target may be represented as for a multiclass

problem but with only two labels A binary decision function is represented as a 1d array

Semantically one class is often considered the “positive” class Unless otherwise specified eg using pos_label

in evaluation metrics we consider the class label with the greater value numerically or lexicographically as

the positive class of labels 0 1 1 is the positive class of 1 2 2 is the positive class of ‘no’ ‘yes’ ‘yes’

is the positive class of ‘no’ ‘YES’ ‘no’ is the positive class This affects the output of decision_function for

instance

Note that a dataset sampled from a multiclass y or a continuous y may appear to be binary

674 Chapter 4 Glossary of Common Terms and API (Application Programming Interface) Elements

scikitlearn user guide Release 0213

typeoftarget will return 'binary' for binary input or a similar array with only a single class present

continuous A regression problem where each sample's target is a finite floating point number represented as a 1dimensional array of floats or sometimes ints

typeoftarget will return 'continuous' for continuous input but if the data is all integers it will be identified as 'multiclass'

continuous multioutput

multioutput continuous A regression problem where each sample's target consists of noutputs outputs each one a finite floating point number for a fixed int noutputs 1 in a particular dataset

Continuous multioutput targets are represented as multiple continuous targets horizontally stacked into an array of shapensamples noutputs

typeoftarget will return 'continuousmultioutput' for continuous multioutput input but if the data is all integers it will be identified as 'multiclassmultioutput'

multiclass A classification problem consisting of more than two classes A multiclass target may be represented as a 1dimensional array of strings or integers A 2d column vector of integers ie a single output in multioutput terms is also accepted

We do not officially support other orderable hashable objects as class labels even if estimators may happen to work when given classification targets of such type

For semisupervised classification unlabeled samples should have the special label 1 in y

Within scikitlearn all estimators supporting binary classification also support multiclass classification using OnevsRest by default

ApreprocessingLabelEncoder helps to canonicalize multiclass targets as integers

typeoftarget will return 'multiclass' for multiclass input The user may also want to handle 'binary' input identically to 'multiclass'

multiclass multioutput

multioutput multiclass A classification problem where each sample's target consists of noutputs outputs each a class label for a fixed int noutputs 1 in a particular dataset Each output has a fixed set of available classes and each sample is labelled with a class for each output An output may be binary or multiclass and in the case where all outputs are binary the target is multilabel

Multiclass multioutput targets are represented as multiple multiclass targets horizontally stacked into an array of shapensamples noutputs

XXX For simplicity we may not always support string class labels for multiclass multioutput and integer class labels should be used

multioutput provides estimators which estimate multioutput problems using multiple singleoutput estimators This may not fully account for dependencies among the different outputs which methods natively handling the multioutput case eg decision trees nearest neighbors neural networks may do better

typeoftarget will return 'multiclassmultioutput' for multiclass multioutput input

multilabel A multiclass multioutput target where each output is binary This may be represented as a 2d dense array or sparse matrix of integers such that each column is a separate binary target where positive labels are indicated with 1 and negative labels are usually 1 or 0 Sparse multilabel targets are not supported everywhere that dense multilabel targets are supported

Semantically a multilabel target can be thought of as a set of labels for each sample While not used internally preprocessingMultiLabelBinarizer is provided as a utility to convert from a list of sets

representation to a 2d array or sparse matrix Onehot encoding a multiclass target with preprocessing

LabelBinarizer turns it into a multilabel problem

43 Target Types 675

scikitlearn user guide Release 0213

typeoftarget will return ‘multilabelindicator’ for multilabel input whether sparse or dense

multioutput

multioutput A target where each sample has multiple classificationregression labels See multiclass multioutput

andcontinuous multioutput We do not currently support modelling mixed classification and regression targets

44 Methods

decisionfunction In a fitted classifier oroutlier detector predicts a “soft” score for each sample in relation to each class rather than the “hard” categorical prediction produced by predict Its input is usually only some

observed data X

If the estimator was not already fitted calling this method should raise a exceptionsNotFittedError

Output conventions

binary classification A 1dimensional array where values strictly greater than zero indicate the positive class

ie the last class in classes

multiclass classification A 2dimensional array where the rowwise argmaximum is the predicted class

Columns are ordered according to classes

multilabel classification Scikitlearn is inconsistent in its representation of multilabel decision functions

Some estimators represent it like multiclass multioutput ie a list of 2d arrays each with two columns

Others represent it with a single 2d array whose columns correspond to the individual binary classification

decisions The latter representation is ambiguously identical to the multiclass classification format though

its semantics differ it should be interpreted like in the binary case by thresholding at 0

TODO This gist highlights the use of the different formats for multilabel

multioutput classification A list of 2d arrays corresponding to each multiclass decision function

outlier detection A 1dimensional array where a value greater than or equal to zero indicates an inlier

fit Thefit method is provided on every estimator It usually takes some samplesXtargetsyif the model is

supervised and potentially other sample properties such as sampleweight It should

- clear any prior attributes stored on the estimator unless warmstart is used
- validate and interpret any parameters ideally raising an error if invalid
- validate the input data
- estimate and store model attributes from the estimated parameters and provided data and
- return the now fitted estimator to facilitate method chaining

Target Types describes possible formats for y

fitpredict Used especially for unsupervised transductive estimators this fits the model and returns the pre

dictions similar to predict on the training data In clusterers these predictions are also stored in the labels

attribute and the output of fitpredictX is usually equivalent to fitXpredictX The pa

rameters to fitpredict are the same as those to fit

fittransform A method on transformers which fits the estimator and returns the transformed training

data It takes parameters as in fitand its output should have the same shape as calling fitX

transformX There are nonetheless rare cases where fittransformX andfitX

transformX do not return the same value wherein training data needs to be handled differently due

to model blending in stacked ensembles for instance such cases should be clearly documented Transductive

transformers may also provide fittransform but not transform

676 Chapter 4 Glossary of Common Terms and API (Application Programming Interface) Elements

scikitlearn user guide Release 0213

One reason to implement `fit_transform` is that performing `fit` and `transform` separately would be less efficient than together `BaseTransformerMixin` provides a default implementation providing a consistent interface across transformers where `fit_transform` is or is not specialised

Inductive learning – where the goal is to learn a generalised model that can be applied to new data – users should be careful not to apply `fit_transform` to the entirety of a dataset ie training and test data together before further modelling as this results in data leakage

`get_feature_names` Primarily for feature extractors but also used for other transformers to provide string names for each column in the output of the estimator’s `transform` method It outputs a list of strings and may take a list of strings as input corresponding to the names of input columns from which output column names can be generated By default input features are named `x0` `x1`

`get_n_splits` On a CV (Cross-Validation) splitter not an estimator returns the number of elements one would get if iterating through the return value of `split` given the same parameters Takes the same parameters as `split`

`get_params` Gets all parameters and their values that can be set using `set_params` A parameter deep can be used when set to `False` to only return those parameters not including ie not due to indirection via contained estimators

Most estimators adopt the definition from `BaseEstimator` which simply adopts the parameters defined for `init` `Pipeline` among others reimplements `get_params` to declare the estimators named in its steps parameters as themselves being parameters

`partial_fit` Facilitates fitting an estimator in an online fashion Unlike `fit` repeatedly calling `partial_fit` does not clear the model but updates it with respect to the data provided The portion of data provided to `partial_fit` may be called a minibatch Each minibatch must be of consistent shape etc In iterative estimators `partial_fit` often only performs a single iteration

`partial_fit` may also be used for outofcore learning although usually limited to the case where learning can be performed online ie the model is usable after each `partial_fit` and there is no separate processing needed to finalize the model `cluster.Birch` introduces the convention that calling `partial_fit_X` will produce a model that is not finalized but the model can be finalized by calling `partial_fit` ie without passing a further minibatch

Generally estimator parameters should not be modified between calls to `partial_fit` although `partial_fit` should validate them as well as the new minibatch of data In contrast `warmstart` is used to repeatedly fit the same estimator with the same data but varying parameters

`LikeFitPartialFit` should return the estimator object

To clear the model a new estimator should be constructed for instance with `base.clone`

`predict` Makes a prediction for each sample usually only taking `X` as input but see under regressor output conventions below In a classifier or regressor this prediction is in the same target space used in fitting eg one of ‘red’ ‘amber’ ‘green’ if the yin fitting consisted of these strings Despite this even when y passed to `fit` is a list or other arraylike the output of `predict` should always be an array or sparse matrix In a clusterer or outlier detector the prediction is an integer

If the estimator was not already fitted calling this method should raise a `exceptions.NotFittedError`

Output conventions

classifier An array of shape `n_samples` `n_outputs` Multilabel data may be represented as a sparse matrix if a sparse matrix was used in fitting Each element should be one of the values in the classifier’s `classes` attribute

clusterer An array of shape `n_samples` where each value is from 0 to `n_clusters - 1` if the corresponding sample is clustered and `-1` if the sample is not clustered as in `clusterdbSCAN`

outlier detector An array of shape `n_samples` where each value is `1` for an outlier and `0` otherwise

scikitlearn user guide Release 0213

regressor A numeric array of shape nsamples usually float64 Some regressors have extra options in their predict method allowing them to return standard deviation returnstdTrue or covariance returncovTrue relative to the predicted value In this case the return value is a tuple of arrays corresponding to prediction mean std cov as required
predictlogproba The natural logarithm of the output of predictproba provided to facilitate numerical stability

predictproba A method in classifiers and clusterers that are able to return probability estimates for each class cluster Its input is usually only some observed data X

If the estimator was not already fitted calling this method should raise a exceptionsNotFittedError
Output conventions are like those for decisionfunction except in the binary classification case where one column is output for each class while decisionfunction outputs a 1d array For binary and multiclass predictions each row should add to 1

Like other methods predictproba should only be present when the estimator can make probabilistic predictions see duck typing This means that the presence of the method may depend on estimator parameters eg inlinearmodelSGDClassifier or training data eg in modelselectionGridSearchCV and may only appear after fitting

score A method on an estimator usually a predictor which evaluates its predictions on a given dataset and returns a single numerical score A greater return value should indicate better predictions accuracy is used for classifiers and R2 for regressors by default

If the estimator was not already fitted calling this method should raise a exceptionsNotFittedError
Some estimators implement a custom estimatorspecific score function often the likelihood of the data under the model

scoresamples TODO

If the estimator was not already fitted calling this method should raise a exceptionsNotFittedError
setparams Available in any estimator takes keyword arguments corresponding to keys in getparams Each is provided a new value to assign such that calling getparams after setparams will reflect the changed parameters Most estimators use the implementation in baseBaseEstimator which handles nested parameters and otherwise sets the parameter as an attribute on the estimator The method is overridden in pipeline Pipeline and related estimators

split On a CV (Cross-Validation) splitter not an estimator this method accepts parameters Xygroups where all may be optional and returns an iterator over trainidx testidx pairs Each of traintestidx is a 1d integer array with values from 0 from Xshape0 1 of any length such that no values appear in both some trainidx and its corresponding testidx

transform In a transformer transforms the input usually only X into some transformed space conventionally notated as Xt Output is an array or sparse matrix of length nsamples and with number of columns fixed after fitting

If the estimator was not already fitted calling this method should raise a exceptionsNotFittedError

45 Parameters

These common parameter names specifically used in estimator construction see concept parameter sometimes also appear as parameters of functions or nonestimator constructors

classweight Used to specify sample weights when fitting classifiers as a function of the target class Where sampleweight is also supported and given it is multiplied by the classweight contribution Similarly

678 Chapter 4 Glossary of Common Terms and API (Application Programming Interface) Elements

scikitlearn user guide Release 0213

whereclassweight is used in a multioutput including multilabel tasks the weights are multiplied across outputs ie columns of y

By default all samples have equal weight such that classes are effectively weighted by their their prevalence in the training data This could be achieved explicitly with classweightlabel1 1 label2 1 for all class labels

More generally classweight is specified as a dict mapping class labels to weights classlabel weight such that each sample of the named class is given that weight classweightbalanced can be used to give all classes equal weight by giving each sample a weight inversely related to its class's prevalence in the training data nsamples nclasses np bincounty Class weights will be used differently depending on the algorithm for linear models such as linear SVM (Support Vector Machine) or logistic regression the class weights will alter the loss function by weighting the loss of each sample by its class weight For treebased algorithms the class weights will be used for reweighting the splitting criterion Note however that this rebalancing does not take the weight of samples in each class into account For multioutput classification a list of dicts is used to specify weights for each output For example for four class multilabel classification weights should be 0 1 1 1 0 1 1 5 0 1 1 1 0 1 1 1 instead of 11 25 31 41

Theclassweight parameter is validated and interpreted with utilscomputeclasseweight cv Determines a cross validation splitting strategy as used in crossvalidation based routines cv is also available in estimators such as multioutputClassifierChain orcalibration CalibratedClassifierCV which use the predictions of one estimator as training data for another to not overfit the training supervision

Possible inputs for cvare usually

- An integer specifying the number of folds in Kfold cross validation Kfold will be stratified over classes if the estimator is a classifier determined by baseisclassifier and the targets may represent a binary or multiclass but not multioutput classification problem determined by utilsmulticlass typeoftarget

- Acrossvalidation splitter instance Refer to the User Guide for splitters available within Scikitlearn
- An iterable yielding traintest splits

With some exceptions especially where not using cross validation at all is an option the default is 3fold and will change to 5fold in version 022

cvvalues are validated and interpreted with utilscheckcv

kernel TODO

maxiter For estimators involving iterative optimization this determines the maximum number of iterations to be performed in fit Ifmaxiter iterations are run without convergence a exceptions ConvergenceWarning should be raised Note that the interpretation of "a single iteration" is inconsistent across estimators some but not all use it to mean a single epoch ie a pass over every sample in the data FIXME perhaps we should have some common tests about the relationship between ConvergenceWarning and maxiter

memory Some estimators make use of joblibMemory to store partial solutions during fitting Thus when fit is called again those partial solutions have been memoized and can be reused Amemory parameter can be specified as a string with a path to a directory or a joblibMemory instance or an object with a similar interface ie a cache method can be used memory values are validated and interpreted with utilsvalidationcheckmemory

45 Parameters 679

scikitlearn user guide Release 0213

metric As a parameter this is the scheme for determining the distance between two data points See metrics pairwise distances In practice for some algorithms an improper distance metric one that does not obey the triangle inequality such as Cosine Distance may be used

XXX hierarchical clustering uses affinity with this meaning

We also use metric to refer to evaluation metrics but avoid using this sense as a parameter name

ncomponents The number of features which a transformer should transform the input into See components for the special case of affine projection

niternochange Number of iterations with no improvement to wait before stopping the iterative procedure

This is also known as a patience parameter It is typically used with early stopping to avoid stopping too early

njobs This is used to specify how many concurrent processes threads should be used for parallelized routines

Scikitlearn uses one processor for its processing by default although it also makes use of NumPy which may be configured to use a threaded numerical processor library like MKL see FAQ

njobs is an int specifying the maximum number of concurrently running jobs If set to 1 all CPUs are used

If 1 is given no joblib level parallelism is used at all which is useful for debugging Even with njobs

1 parallelism may occur due to numerical processing libraries see FAQ For njobs below 1 ncpus 1

njobs are used Thus for njobs 2 all CPUs but one are used

njobsNone means unset it will generally be interpreted as njobs1 unless the current joblib

Parallel backend context specifies otherwise

The use of njobs based parallelism in estimators varies

- Most often parallelism happens in fitting but sometimes parallelism happens in prediction eg in random forests

- Some parallelism uses a multithreading backend by default some a multiprocessing backend It is possible to override the default backend by using sklearnutilsparallelbackend

- Whether parallel processing is helpful at improving runtime depends on many factors and it's usually a good idea to experiment rather than assuming that increasing the number of jobs is always a good thing It can be highly detrimental to performance to run multiple copies of some estimators or functions in parallel

Nested uses of njobs based parallelism with the same backend will result in an exception So

GridSearchCVOneVsRestClassifierSVC njobs2 njobs2 won't work

When njobs is not 1 the estimator being parallelized must be picklable This means for instance that

lambdas cannot be used as estimator parameters

randomstate Whenever randomization is part of a Scikitlearn algorithm a randomstate parameter may be provided to control the random number generator used Note that the mere presence of randomstate doesn't mean that randomization is always used as it may be dependent on another parameter eg shuffle being set

randomstate 's value may be

None default Use the global random state from numpyrandom

An integer Use a new random number generator seeded by the given integer To make a randomized algorithm deterministic ie running it multiple times will produce the same result an arbitrary integer

randomstate can be used However it may be worthwhile checking that your results are stable across a number of different distinct random seeds Popular integer random seeds are 0 and 42

AnumpyrandomRandomState instance Use the provided random state only affecting other users of the same random state instance Calling fit multiple times will reuse the same instance and will produce different results

680 Chapter 4 Glossary of Common Terms and API (Application Programming Interface) Elements

scikitlearn user guide Release 0213

utilscheckrandomstate is used internally to validate the input randomstate and return a RandomState instance

scoring Specifies the score function to be maximized usually by cross validation or – in some cases – multiple score functions to be reported The score function can be a string accepted by metricsgetscorer or a callable scorer not to be confused with an evaluation metric as the latter have a more diverse API (Application Programming Interface) scoring may also be set to None in which case the estimator’s score method is used See The scoring parameter defining model evaluation rules in the User Guide

Where multiple metrics can be evaluated scoring may be given either as a list of unique strings or a dict with names as keys and callables as values Note that this does not specify which score function is to be maximised and another parameter such as refit may be used for this purpose

The scoring parameter is validated and interpreted using metricscheckscoring

verbose Logging is not handled very consistently in Scikitlearn at present but when it is provided as an option the verbose parameter is usually available to choose no logging set to False Any True value should enable some logging but larger integers eg above 10 may be needed for full verbosity Verbose logs are usually printed to Standard Output Estimators should not produce any output on Standard Output with the default verbose setting

warmstart When fitting an estimator repeatedly on the same dataset but for multiple parameter values such as to find the value maximizing performance as in grid search it may be possible to reuse aspects of the model learnt from the previous parameter value saving time When warmstart is true the existing fitted model attributes are used to initialise the new model in a subsequent call to fit

Note that this is only applicable for some models and some parameters and even some orders of parameter values For example warmstart may be used when building random forests to add more trees to the forest increasing nestimators but not to reduce their number

partialfit also retains the model between calls but differs with warmstart the parameters change and the data is more or less constant across calls to fit with partialfit the minibatch of data changes and model parameters stay fixed

There are cases where you want to use warmstart to fit on different but closely related data For example one may initially fit to a subset of the data then finetune the parameter search on the full dataset For classification all data in a sequence of warmstart calls to fit must include samples from each class

46 Attributes

See concept attribute

classes A list of class labels known to the classifier mapping each label to a numerical index used in the model representation our output For instance the array output from predictproba has columns aligned with classes For multioutput classifiers classes should be a list of lists with one class listing for each output For each output the classes should be sorted numerically or lexicographically for strings classes and the mapping to indices is often managed with preprocessingLabelEncoder

components An affine transformation matrix of shape ncomponents nfeatures used in many linear transformers where ncomponents is the number of output features and nfeatures is the number of input features

See also components which is a similar attribute for linear predictors

coef The weightcoefficient matrix of a generalised linear model predictor of shapenfeatures for binary classification and singleoutput regression nclasses nfeatures for multiclass classification and ntargets nfeatures for multioutput regression Note this does not include the intercept or bias term which is stored in intercept

46 Attributes 681

scikitlearn user guide Release 0213

When available featureimportances is not usually provided as well but can be calculated as the norm of each feature's entry in coef

See also components which is a similar attribute for linear transformers

embedding An embedding of the training data in manifold learning estimators with shape nsamples

ncomponents identical to the output of fittransform See also labels

niter The number of iterations actually performed when fitting an iterative estimator that may stop upon convergence See also maxiter

featureimportances A vector of shape nfeatures available in some predictors to provide a relative measure of the importance of each feature in the predictions of the model

labels A vector containing a cluster label for each sample of the training data in clusterers identical to the output

offitpredict See also embedding

47 Data and sample properties

See concept sample property

groups Used in cross validation routines to identify samples which are correlated Each value is an identifier such that in a supporting CV (Cross-Validation) splitter samples from some groups value may not appear in both a training set and its corresponding test set See Crossvalidation iterators for grouped data

sampleweight A relative weight for each sample Intuitively if all weights are integers a weighted model or score should be equivalent to that calculated when repeating the sample the number of times specified in the weight Weights may be specified as floats so that sample weights are usually equivalent up to a constant positive scaling factor

FIXME Is this interpretation always the case in practice We have no common tests

Some estimators such as decision trees support negative weights FIXME This feature or its absence may not be tested or documented in many estimators

This is not entirely the case where other parameters of the model consider the number of samples in a region as withminsamples inclusterDBSCAN In this case a count of samples becomes to a sum of their weights

In classification sample weights can also be specified as a function of class with the classweight estimator parameter

XDenotes data that is observed at training and prediction time used as independent variables in learning The notation is uppercase to denote that it is ordinarily a matrix see rectangular When a matrix each sample may be represented by a feature vector or a vector of precomputed dissimilarity with each training sample Xmay also not be a matrix and may require a feature extractor or apairwise metric to turn it into one before learning a model

Xt Shorthand for "transformed X"

y

YDenotes data that may be observed at training time as the dependent variable in learning but which is unavailable at prediction time and is usually the target of prediction The notation may be uppercase to denote that it is a matrix representing multioutput targets for instance but usually we use yand sometimes do so even when multiple outputs are assumed

682 Chapter 4 Glossary of Common Terms and API (Application Programming Interface) Elements

CHAPTER
FIVE
EXAMPLES
51 Miscellaneous examples
Miscellaneous and introductory examples for scikitlearn
Note [Click here to download the full example code](#)
511 Compact estimator representations
This example illustrates the use of the printchangedonly global parameter
Setting printchangedonly to True will alterate the representation of estimators to only show the parameters that have been set to nondefault values This can be used to have more compact representations
Out
Default representation
LogisticRegressionC10 classweightNone dualFalse fitinterceptTrue
interceptscaling1 l1ratioNone maxiter100
multiclasswarn njobsNone penaltyl1
randomstateNone solverwarn tol00001 verbose0
warmstartFalse
With changedonly option
LogisticRegressionpenaltyl1
printdoc
from sklearnlinearmodel import LogisticRegression
from sklearn import setconfig
lr LogisticRegressionpenaltyl1
printDefault representation
printlr
LogisticRegressionC10 classweightNone dualFalse fitinterceptTrue
interceptscaling1 l1ratioNone maxiter100
683

scikitlearn user guide Release 0213
multiclasswarn njobsNone penaltyl1
randomstateNone solverwarn tol00001 verbose0
warmstartFalse
setconfigprintchangedonlyTrue
printnWith changedonly option
printlr
LogisticRegressionpenaltyl1
Total running time of the script 0 minutes 0003 seconds
Note [Click here to download the full example code](#)

512 Isotonic Regression
An illustration of the isotonic regression on generated data The isotonic regression finds a nondecreasing approximation of a function while minimizing the mean squared error on the training data The benefit of such a model is that it does not assume any form for the target function such as linearity For comparison a linear regression is also presented
printdoc
684 Chapter 5 Examples

scikitlearn user guide Release 0213
Author Nelle Varoquaux nellevaroquauxgmailcom
Alexandre Gramfort alexandregramfortinriafr
License BSD (Binary Search Decision)

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.collections import LineCollection
from sklearn.linear_model import LinearRegression
from sklearn.isotonic import IsotonicRegression
from sklearn.utils import check_random_state
n = 100
x = np.arange(n)
rs = check_random_state(0)
y = rs.randint(50, 50, size=n)
nplog1pnp.arange(n)
```

Fit IsotonicRegression and LinearRegression models
ir = IsotonicRegression
y = ir.fit_transform(x, y)
lr = LinearRegression
lr.fit(x, np.newaxis * y) # x needs to be 2d for LinearRegression

```
# Plot result
segments = [y[i] for i in range(n)]
lc = LineCollection(segments, zorder=0)
lc.set_array(np.ones(n))
lc.set_linewidths(np.full(n, 0.5))
fig = plt.figure()
plt.plot(x, y, 'r', markersize=12)
plt.plot(x, lr.predict(x), 'b', markersize=12)
plt.gca().add_collection(lc)
plt.legend(['Data', 'Isotonic Fit', 'Linear Fit'], loc='lower right')
plt.title('Isotonic regression')
plt.show
```

Total running time of the script: 0 minutes 00.56 seconds

Note: Click [here](#) to download the full example code

513 Face completion with a multi-output estimator

This example shows the use of a multi-output estimator to complete images. The goal is to predict the lower half of a face given its upper half.

The first column of images shows true faces. The next columns illustrate how extremely randomized trees, k-nearest neighbors, linear regression, and ridge regression complete the lower half of those faces.

51 Miscellaneous examples 685

scikitlearn user guide Release 0213
Out
downloading Olivetti faces from httpsndownloaderfigsharecomfiles5976027 to
↩→homecircleciscikitlearndata
686 Chapter 5 Examples

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import fetcholivettifaces
from sklearn.utils.validation import check_random_state
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import RidgeCV
Load the faces datasets
data = fetcholivettifaces
targets = data.target
data = data.images.reshape(len(data.images), 1)
train = data[:30]
test = data[30:]
Test on independent people
Test on a subset of people
nfaces = 5
rng = check_random_state(4)
faceids = rng.randint(test.shape[0], size=nfaces)
test = test[faceids]
npixels = data.shape[1]
Upper half of the faces
Xtrain = train[npixels:]
Lower half of the faces
ytrain = train[:, npixels:]
Xtest = test[npixels:]
ytest = test[:, npixels:]
Fit estimators
ESTIMATORS
Extra trees ExtraTreesRegressor(n_estimators=10, max_features=32)
random_state=0
Knn KNeighborsRegressor
Linear regression LinearRegression
Ridge RidgeCV

ytest_predict = dict
for name, estimator in ESTIMATORS.items():
    estimator.fit(Xtrain, ytrain)
ytest_predict[name] = estimator.predict(Xtest)
Plot the completed faces
imageshape = (64, 64)
ncols = 1
lenESTIMATORS
plt.figure(figsize=(2, ncols), dpi=226, nfaces=nfaces)
plt.suptitle('Face completion with multi-output estimators', size=16)
for i in range(nfaces):
    trueface = np.hstack(Xtest[i, :])
51 Miscellaneous examples 687
```

scikitlearn user guide Release 0213

```
ifi
sub pltsubplotnfaces ncols i ncols 1
else
sub pltsubplotnfaces ncols i ncols 1
titletrue faces
subaxisoff
subimshowtruefacereshapeimageshape
cmapppltcmgray
interpolationnearest
forj estinenumeratesortedESTIMATORS
completedface nphstackXtesti ytestpredictesti
ifi
sub pltsubplotnfaces ncols i ncols 2 j
else
sub pltsubplotnfaces ncols i ncols 2 j
titleest
subaxisoff
subimshowcompletedfacereshapeimageshape
cmapppltcmgray
interpolationnearest
pltshow
```

Total running time of the script 0 minutes 3749 seconds

Note Click here to download the full example code

514 Multilabel classification

This example simulates a multilabel document classification problem The dataset is generated randomly based on the following process

- pick the number of labels n Poissonnlabels
- n times choose a class c c Multinomialtheta
- pick the document length k Poissonlength
- k times choose a word w Multinomialthetac

In the above process rejection sampling is used to make sure that n is more than 2 and that the document length is never zero Likewise we reject classes which have already been chosen The documents that are assigned to both classes are plotted surrounded by two colored circles

The classification is performed by projecting to the first two principal components found by PCA (Principal Component Analysis) and CCA for visualisation purposes followed by using the sklearnmulticlassOneVsRestClassifier metaclassifier using two SVCs with linear kernels to learn a discriminative model for each class Note that PCA (Principal Component Analysis) is used to perform an unsupervised dimensionality reduction while CCA is used to perform a supervised one Note in the plot “unlabeled samples” does not mean that we don’t know the labels as in semisupervised learning but that the samples simply do nothave a label

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_multilabel_classification
from sklearn.multiclass import OneVsRestClassifier
from sklearn.svm import SVC (Support Vector Classification)
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.cross_decomposition import CCA
def plot_hyperplane_clf(minx, maxx, linestyle, label):
    # get the separating hyperplane
    w = clf.coef_0
    a = w[0], w[1]
    xx = np.linspace(minx, maxx, 5) # make sure the line is long enough
    yy = -a[0] / a[1] * xx + w[2] / a[1]
    plt.plot(xx, yy, linestyle, label)
def plot_subfigure(X, Y, subplot_title, transform):
    if transform == 'pca':
        X = PCA(n_components=2).fit_transform(X)
    elif transform == 'cca':
        X = CCA(n_components=2).fit(X, Y).transform(X)
51 Miscellaneous examples 689
```

scikitlearn user guide Release 0213

```
else
raiseValueError
minx npminX 0
maxx npmaxX 0
miny npminX 1
maxy npmaxX 1
classif OneVsRestClassifierSVCKernellinear
classiffitX Y
pltsubplot2 2 subplot
plttitletitle
zeroclass npwhereY 0
oneclass npwhereY 1
pltscatterX 0 X 1 s40 cgray edgecolors0 0 0
pltscatterXzeroclass 0 Xzeroclass 1 s160 edgecolorsb
facecolorsnone linewidths2 labelClass 1
pltscatterXoneclass 0 Xoneclass 1 s80 edgecolorssorange
facecolorsnone linewidths2 labelClass 2
plothyperplaneclassestestimators0 minx maxx k
Boundary nfor class 1
plothyperplaneclassestestimators1 minx maxx k
Boundary nfor class 2
pltxticks
pltyticks
pltxlimminx 5 maxx maxx 5 maxx
pltylimminy 5 maxy maxy 5 maxy
ifsubplot 2
pltxlabelFirst principal component
pltylabelSecond principal component
pltlegendlocupper left
pltfigurefigsize8 6
X Y makemultilabelclassificationnnclasses2 nlabels1
allowunlabeledTrue
randomstate1
plotsubfigureX Y 1 With unlabeled samples CCA cca
plotsubfigureX Y 2 With unlabeled samples PCA (Principal Component Analysis) pca
X Y makemultilabelclassificationnnclasses2 nlabels1
allowunlabeledFalse
randomstate1
plotsubfigureX Y 3 Without unlabeled samples CCA cca
plotsubfigureX Y 4 Without unlabeled samples PCA (Principal Component Analysis) pca
pltsubplotsadjust04 02 97 94 09 2
pltshow
Total running time of the script 0 minutes 0093 seconds
690 Chapter 5 Examples
```


scikitlearn user guide Release 0213

Note Click here to download the full example code

515 Comparing anomaly detection algorithms for outlier detection on toy datasets

This example shows characteristics of different anomaly detection algorithms on 2D datasets. Datasets contain one or two modes/regions of high density to illustrate the ability of algorithms to cope with multimodal data.

For each dataset, 15% of samples are generated as random uniform noise. This proportion is the value given to the `nu` parameter of the `OneClassSVM` and the contamination parameter of the other outlier detection algorithms. Decision boundaries between inliers and outliers are displayed in black except for Local Outlier Factor (LOF) as it has no predict method to be applied on new data when it is used for outlier detection.

`sklearnsvmOneClassSVM` is known to be sensitive to outliers and thus does not perform very well for outlier detection. This estimator is best suited for novelty detection when the training set is not contaminated by outliers. That said, outlier detection in high dimension or without any assumptions on the distribution of the inlying data is very challenging and a `OneClass SVM` (Support Vector Machine) might give useful results in these situations depending on the value of its hyperparameters.

`sklearn covariance EllipticEnvelope` assumes the data is Gaussian and learns an ellipse. It thus degrades when the data is not unimodal. Notice however that this estimator is robust to outliers.

`sklearn ensemble IsolationForest` and `sklearn neighbors LocalOutlierFactor` seem

to perform reasonably well for multimodal data sets. The advantage of `sklearn neighbors`

`LocalOutlierFactor` over the other estimators is shown for the third data set where the two modes have different densities. This advantage is explained by the local aspect of LOF meaning that it only compares the score of abnormality of one sample with the scores of its neighbors.

Finally, for the last data set, it is hard to say that one sample is more abnormal than another sample as they are uniformly distributed in a hypercube. Except for the `sklearnsvmOneClassSVM` which overfits a little, all estimators present decent solutions for this situation. In such a case, it would be wise to look more closely at the scores of abnormality of the samples as a good estimator should assign similar scores to all the samples.

While these examples give some intuition about the algorithms, this intuition might not apply to very high dimensional data.

Finally, note that parameters of the models have been here handpicked but that in practice they need to be adjusted. In the absence of labelled data, the problem is completely unsupervised so model selection can be a challenge.

51 Miscellaneous examples 691

scikitlearn user guide Release 0213
Author Alexandre Gramfort alexandregamfortinriafr
Albert Thomas albertthomastelecomparistechfr
License BSD (Binary Search Decision) 3 clause
import time
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
692 Chapter 5 Examples

```
scikitlearn user guide Release 0213
from sklearn import svm
from sklearn.datasets import makemoons makeblobs
from sklearn.covariance import EllipticEnvelope
from sklearn.ensemble import IsolationForest
from sklearn.neighbors import LocalOutlierFactor
printdoc
matplotlib.rcParams['contour.negative_linestyle'] = 'solid'
Example settings
nsamples = 300
outliersfraction = 0.15
noutliers = int(outliersfraction * nsamples)
ninliers = nsamples - noutliers
define outlier anomaly detection methods to be compared
anomaly_algorithms
Robust covariance EllipticEnvelope contamination_outliersfraction
OneClass SVM (Support Vector Machine) svmOneClassSVM nu_outliersfraction kernel_rbf
gamma=0.1
Isolation Forest IsolationForest behaviour new
contamination_outliersfraction
random_state=42
Local Outlier Factor LocalOutlierFactor
n_neighbors=35 contamination_outliersfraction
Define datasets
blobs_params = dict(random_state=0, nsamples=ninliers, nfeatures=2)
datasets
makeblobs_centers=[0, 0, 0], cluster_std=0.5
blobs_params=0
makeblobs_centers=[2, 2, 2], cluster_std=[0.5, 0.5]
blobs_params=0
makeblobs_centers=[2, 2, 2], cluster_std=[15, 3]
blobs_params=0
4makemoons nsamples nsamples noise=0.5 random_state=0
np.array([0.5, 0.25])
14np.random.RandomState(42).randn(samples=2, size=0.5)
Compare given classifiers under given settings
xx, yy = np.meshgrid(np.linspace(-7, 7, 150),
np.linspace(-7, 7, 150))
plt.figure(figsize=(12, 12))
plt.subplot(2, 2, 1)
plt.subplot(2, 2, 2)
plt.subplot(2, 2, 3)
plt.subplot(2, 2, 4)
plt.tight_layout()
hspace=0.1
plotnum = 1
rng = np.random.RandomState(42)
for dataset, X in enumerate(datasets):
    Add outliers
    X = np.concatenate((X, rng.uniform(-6, 6, (n_outliers, 2))))
    for name, algorithm in anomaly_algorithms.items():
        51 Miscellaneous examples 693
```

```

scikitlearn user guide Release 0213
t0 = time
algorithmfitX
t1 = time
plt.subplot(2, 2, 1)
plt.title('Local Outlier Factor')
plt.xlabel('Algorithm')
plt.ylabel('Predicted')
plt.grid(True)
plt.show()

```

Total running time of the script: 0 minutes 34.91 seconds
 Note: Click [here](#) to download the full example code

5.16 The Johnson-Lindenstrauss bound for embedding with random projections

The Johnson-Lindenstrauss lemma states that any high dimensional dataset can be randomly projected into a lower dimensional Euclidean space while controlling the distortion in the pairwise distances.

Theoretical bounds

The distortion introduced by a random projection is asserted by the fact that is defining an embedding with good probability as defined by

$$1 - \frac{1}{n} \sum_{i=1}^n \left| \frac{\|u_i - v_i\|^2}{\|u_i - v_i\|^2} - \frac{\|u_i - v_i\|^2}{\|u_i - v_i\|^2} \right| \leq \frac{1}{n} \sum_{i=1}^n \left| \frac{\|u_i - v_i\|^2}{\|u_i - v_i\|^2} - \frac{\|u_i - v_i\|^2}{\|u_i - v_i\|^2} \right|$$

Where u and v are any rows taken from a dataset of shape $n \times m$ and p is a projection by a random Gaussian $N(0, 1)$ matrix with shape $n \times m$ or a sparse Achlioptas matrix.

The minimum number of components to guarantees the embedding is given by

$$m \geq \frac{4 \log(n) \log(1/\epsilon)}{\epsilon^2}$$

694 Chapter 5 Examples

scikitlearn user guide Release 0213

The first plot shows that with an increasing number of samples `nsamples` the minimal number of dimensions `ncomponents` increased logarithmically in order to guarantee an `epembedding`

The second plot shows that an increase of the admissible distortion `eps` allows to reduce drastically the minimal number of dimensions `ncomponents` for a given number of samples `nsamples`

Empirical validation

We validate the above bounds on the digits dataset or on the 20 newsgroups text document TFIDF word frequencies dataset

- for the digits dataset some 8x8 gray level pixels data for 500 handwritten digits pictures are randomly projected to spaces for various larger number of dimensions `ncomponents`
- for the 20 newsgroups dataset some 500 documents with 100k features in total are projected using a sparse random matrix to smaller euclidean spaces with various values for the target number of dimensions `ncomponents`

The default dataset is the digits dataset To run the example on the twenty newsgroups dataset pass the `-twentynewsgroups` command line argument to this script

For each value of `ncomponents` we plot

- 2D distribution of sample pairs with pairwise distances in original and projected spaces as x and y axis respectively
- 1D histogram of the ratio of those distances projected / original

We can see that for low values of `ncomponents` the distribution is wide with many distorted pairs and a skewed distribution due to the hard limit of zero ratio on the left as distances are always positives while for larger values of `ncomponents` the distortion is controlled and the distances are well preserved by the random projection

Remarks

According to the JL lemma projecting 500 samples without too much distortion will require at least several thousands dimensions irrespective of the number of features of the original dataset

Hence using random projections on the digits dataset which only has 64 features in the input space does not make sense it does not allow for dimensionality reduction in this case

On the twenty newsgroups on the other hand the dimensionality can be decreased from 56436 down to 10000 while reasonably preserving pairwise distances

51 Miscellaneous examples 695

scikitlearn user guide Release 0213

-

51 Miscellaneous examples 697

scikitlearn user guide Release 0213

-

51 Miscellaneous examples 699

scikitlearn user guide Release 0213

- 700 Chapter 5 Examples

scikitlearn user guide Release 0213

-

51 Miscellaneous examples 701

scikitlearn user guide Release 0213

•

Out

Embedding 500 samples with dim 64 using various random projections

Projected 500 samples from 64 to 300 in 0016s

Random matrix with size 0028MB

Mean distances rate 097 008

Projected 500 samples from 64 to 1000 in 0048s

Random matrix with size 0096MB

Mean distances rate 099 005

Projected 500 samples from 64 to 10000 in 0594s

Random matrix with size 0964MB

Mean distances rate 101 001

printdoc

import sys

from time import time

import numpy as np

import matplotlib

import matplotlib.pyplot as plt

from distutils.version import LooseVersion

from sklearn.randomprojection import johnsonlindenstraussmindim

51 Miscellaneous examples 703

```
scikitlearn user guide Release 0213
from sklearnrandomprojection import SparseRandomProjection
from sklearndatasets import fetch20newsgroupsvectorized
from sklearndatasets import loaddigits
from sklearnmetricspairwise import euclidean_distances
    normed is being deprecated in favor of density in histograms
if LooseVersion(matplotlib.__version__) < 2.1:
    densityparam = density = True
else:
    densityparam = normed = True
Part 1 plot the theoretical dependency between n_components_min and
n_samples
    range of admissible distortions
epsrange = linspace(0.1, 0.99, 5)
colors = plt.cm.Blues(nlinspace(0.3, 1.0, len(epsrange)))
    range of number of samples observation to embed
n_samplesrange = logspace(1.9, 9)
plt.figure
for eps, color in zip(epsrange, colors):
    minn_components, johnson_lindenstrauss_min_dim, n_samplesrange, eps, eps
    plt.loglog(n_samplesrange, minn_components, color=color)
    plt.legend(eps, 0.1f, eps, for eps in epsrange, loc='lower right')
    plt.xlabel('Number of observations to embed')
    plt.ylabel('Minimum number of dimensions')
    plt.title('Johnson-Lindenstrauss bounds: n_samples vs n_components')
    range of admissible distortions
epsrange = linspace(0.01, 0.99, 100)
    range of number of samples observation to embed
n_samplesrange = logspace(2.6, 5)
colors = plt.cm.Blues(nlinspace(0.3, 1.0, len(n_samplesrange)))
plt.figure
for n_samples, color in zip(n_samplesrange, colors):
    minn_components, johnson_lindenstrauss_min_dim, n_samples, eps, epsrange
    plt.semilogy(epsrange, minn_components, color=color)
    plt.legend(n_samples, d, n for n in n_samplesrange, loc='upper right')
    plt.xlabel('Distortion')
    plt.ylabel('Minimum number of dimensions')
    plt.title('Johnson-Lindenstrauss bounds: n_components vs eps')
Part 2 perform sparse random projection of some digits images which are
quite low dimensional and dense or documents of the 20 newsgroups dataset
which is both high dimensional and sparse
if twenty_newsgroups is None:
    Need an internet connection hence not enabled by default
data = fetch20newsgroups_vectorized(data=500)
else:
    data = load_digits(data=500)
```

```
scikitlearn user guide Release 0213
nsamples nfeatures  datashape
printEmbedding dsamples with dim dusing various random projections
    nsamples nfeatures
ncomponentsrange  nparray300 1000 10000
dists  euclidean distancesdata squaredTrueravel
    select only nonidentical samples pairs
nonzero  dists  0
dists  distsnonzero
for ncomponents in ncomponentsrange
    t0  time
    rp  SparseRandomProjectionncomponentsncomponents
    projecteddata  rpfitttransformdata
    printProjected dsamples from dtodin03fs
    nsamples nfeatures ncomponents time  t0
    if hasattr rp  components
        nbytes  rpcomponentsdata.nbytes
        nbytes  rpcomponentsindices.nbytes
        printRandom matrix with size 03fMB  nbytes  1e6
        projecteddists  euclidean distances
        projecteddata  squaredTrueravelnonzero
        pltfigure
        plthexbindists projecteddists gridsizesize100 cmappltcmPuBu
        pltxlabelPairwise squared distances in original space
        pltylabelPairwise squared distances in projected space
        plttitlePairwise distances distribution for ncomponents d
        ncomponents
        cb  pltcolorbar
        cbsetlabelSample pairs counts
        rates  projecteddists  dists
        printMean distances rate 02f02f
        npmeanrates npstdrates
        pltfigure
        plthistrates bins50 range0 2 edgecolor k densityparam
        pltxlabelSquared distances rate projected  original
        pltylabelDistribution of samples pairs
        plttitleHistogram of pairwise distance rates for ncomponents d
        ncomponents
        TODO compute the expected value of eps and add them to the previous plot
        as vertical lines  region
        pltshow
Total running time of the script  0 minutes 1837 seconds
Note Click here to download the full example code
51 Miscellaneous examples 705
```

517 Comparison of kernel ridge regression and SVR (Support Vector Regression)

Both kernel ridge regression KRR and SVR (Support Vector Regression) learn a nonlinear function by employing the kernel trick ie they learn a linear function in the space induced by the respective kernel which corresponds to a nonlinear function in the original space They differ in the loss functions ridge versus epsiloninsensitive loss In contrast to SVR (Support Vector Regression) fitting a KRR can be done in closedform and is typically faster for mediumsized datasets On the other hand the learned model is nonsparse and thus slower than SVR (Support Vector Regression) at predictiontime This example illustrates both methods on an artificial dataset which consists of a sinusoidal target function and strong noise added to every fifth datapoint The first figure compares the learned model of KRR and SVR (Support Vector Regression) when both complexityregularization and bandwidth of the RBF (Radial Basis Function) kernel are optimized using gridsearch The learned functions are very similar however fitting KRR is approx seven times faster than fitting SVR (Support Vector Regression) both with gridsearch However prediction of 100000 target values is more than tree times faster with SVR (Support Vector Regression) since it has learned a sparse model using only approx 13 of the 100 training datapoints as support vectors The next figure compares the time for fitting and prediction of KRR and SVR (Support Vector Regression) for different sizes of the training set Fitting KRR is faster than SVR (Support Vector Regression) for medium sized training sets less than 1000 samples however for larger training sets SVR (Support Vector Regression) scales better With regard to prediction time SVR (Support Vector Regression) is faster than KRR for all sizes of the learned sparse solution Note that the degree of sparsity and thus the prediction time depends on the parameters epsilon and C of the SVR (Support Vector Regression)

scikitlearn user guide Release 0213

-

51 Miscellaneous examples 707

scikitlearn user guide Release 0213

•

Out

SVR (Support Vector Regression) complexity and bandwidth selected and model fitted in 0389 s

KRR complexity and bandwidth selected and model fitted in 0175 s

Support vector ratio 0320

SVR (Support Vector Regression) prediction for 100000 inputs in 0117 s

KRR prediction for 100000 inputs in 0141 s

Authors Jan Hendrik Metzen jhminformatikunibremende

License BSD (Binary Search Decision) 3 clause

import time

import numpy as np

from sklearnsvm import SVR (Support Vector Regression)

from sklearnmodelselection import GridSearchCV

from sklearnmodelselection import learningcurve

from sklearnkernelridge import KernelRidge

import matplotlib.pyplot as plt

708 Chapter 5 Examples

scikitlearn user guide Release 0213
rng nprandomRandomState0

Generate sample data
X 5rnggrand10000 1
y npsinXravel
Add noise to targets
y5 3 05 rnggrandXshape0 5
Xplot nplinspace0 5 100000 None

Fit regression model
trainsize 100
svr GridSearchCVSVRkernelrbf gamma01 cv5
paramgridC 1e0 1e1 1e2 1e3
gamma nplogspace2 2 5
kr GridSearchCVKernelRidgekernelrbf gamma01 cv5
paramgridalpha 1e0 01 1e2 1e3
gamma nplogspace2 2 5
t0 timetime
svrfitXtrainsize ytrainsize
svrfit timetime t0
printSVR complexity and bandwidth selected and model fitted in 3fs
svrfit
t0 timetime
krfitXtrainsize ytrainsize
krfit timetime t0
printKRR complexity and bandwidth selected and model fitted in 3fs
krfit
svratio svrbestestimatorsupportshape0 trainsize
printSupport vector ratio 3f svratio
t0 timetime
ysvr svrpredictXplot
svrpredict timetime t0
printSVR prediction for dinputs in 3fs
Xplotshape0 svrpredict
t0 timetime
ykr krpredictXplot
krpredict timetime t0
printKRR prediction for dinputs in 3fs
Xplotshape0 krpredict

Look at the results
svind svrbestestimatorsupport
pltscatterXsvind ysvind cr s50 labelSVR support vectors
zorder2 edgecolors0 0 0
pltscatterX100 y100 ck labeldata zorder1
edgecolors0 0 0
51 Miscellaneous examples 709

```
scikitlearn user guide Release 0213
pltplotXplot ysvr cr
labelSVR fit 3fs predict 3fs svrfit svrpredict
pltplotXplot ykr cg
labelKRR fit 3fs predict 3fs krfit krpredict
pltxlabeldata
pltylabeltarget
plttitleSVR versus Kernel Ridge
pltlegend
Visualize training and prediction time
pltfigure
Generate sample data
X 5rngrand10000 1
y npsinXravel
y5 3 05 rngrandXshape0 5
sizes nplogspace1 4 7astypenpint
forname estimator inKRR KernelRidgekernelrbf alpha01
gamma10
SVR (Support Vector Regression) SVRkernelrbf C1e1 gamma10items
traintime
testtime
fortraintestsize insizes
t0 timetime
estimatorfitXtraintestsize ytraintestsize
traintimeappendtimetime t0
t0 timetime
estimatorpredictXplot1000
testtimeappendtimetime t0
pltplotsizes traintime o colorr ifname SVR (Support Vector Regression) elseg
labelstrain name
pltplotsizes testtime o colorr ifname SVR (Support Vector Regression) elseg
labelstest name
pltyscalelog
pltyscalelog
pltxlabelTrain size
pltylabelTime seconds
plttitleExecution Time
pltlegendlocbest
Visualize learning curves
pltfigure
svr SVRkernelrbf C1e1 gamma01
kr KernelRidgekernelrbf alpha01 gamma01
trainsizes trainscoressvr testscoressvr
learningcurvesvr X100 y100 trainsizesnplinspace01 1 10
scoringnegmeansquarederror cv10
trainsizesabs trainscoreskr testscoreskr
learningcurvekr X100 y100 trainsizesnplinspace01 1 10
scoringnegmeansquarederror cv10
pltplottrainsizes testscoressvrmean1 o colorr
labelSVR
pltplottrainsizes testscoreskrmean1 o colorg
710 Chapter 5 Examples
```

scikitlearn user guide Release 0213

labelKRR

pltxlabelTrain size

pltylabelMean Squared Error

plttitleLearning curves

pltlegendlocbest

pltshow

Total running time of the script 0 minutes 13067 seconds

Note Click here to download the full example code

518 Explicit feature map approximation for RBF (Radial Basis Function) kernels

An example illustrating the approximation of the feature map of an RBF (Radial Basis Function) kernel

It shows how to use RBFSampler andNystroem to approximate the feature map of an RBF (Radial Basis Function) kernel for classification

with an SVM (Support Vector Machine) on the digits dataset Results using a linear SVM (Support Vector Machine) in the original space a linear SVM

imate mappings and using a kernelized SVM (Support Vector Machine) are compared Timings and accuracy for varying amounts of Monte

Carlo samplings in the case of RBFSampler which uses random Fourier features and different sized subsets of the

training set for Nystroem for the approximate mapping are shown

Please note that the dataset here is not large enough to show the benefits of kernel approximation as the exact SVM (Support Vector Machine) is

still reasonably fast

Sampling more dimensions clearly leads to better classification results but comes at a greater cost This means there

is a tradeoff between runtime and accuracy given by the parameter ncomponents Note that solving the Linear

SVM (Support Vector Machine) and also the approximate kernel SVM (Support Vector Machine) could be greatly accelerated by using stochastic gra

sklearnlinearmodelSGDClassifier This is not easily possible for the case of the kernelized SVM (Support Vector Machine)

The second plot visualized the decision surfaces of the RBF (Radial Basis Function) kernel SVM (Support Vector Machine) and the linear SVM (Supp

maps The plot shows decision surfaces of the classifiers projected onto the first two principal components of the data

This visualization should be taken with a grain of salt since it is just an interesting slice through the decision surface

in 64 dimensions In particular note that a datapoint represented as a dot does not necessarily be classified into the

region it is lying in since it will not lie on the plane that the first two principal components span

The usage of RBFSampler andNystroem is described in detail in Kernel Approximation

51 Miscellaneous examples 711

scikitlearn user guide Release 0213

-
-

712 Chapter 5 Examples

```
scikitlearn user guide Release 0213
printdoc
Author Gael Varoquaux gael dot varoquaux at normalesup dot org
Andreas Mueller amuelleraisunibonnde
License BSD (Binary Search Decision) 3 clause
Standard scientific Python imports
import matplotlib.pyplot as plt
import numpy as np
from time import time

Import datasets classifiers and performance metrics
from sklearn import datasets svm pipeline
from sklearn.kernelapproximation import RBFSampler
Nystroem
from sklearn.decomposition import PCA (Principal Component Analysis)
The digits dataset
digits = datasets.load_digits(nclass=9)
To apply an classifier on this data we need to flatten the image to
turn the data in a samples feature matrix
n_samples = len(digits.data)
data = digits.data.reshape(-1, 16)
data = data.mean(axis=0)
We learn the digits on the first half of the digits
data_train, target_train = data[:n_samples/2], target[:n_samples/2]
digit_target_train = target[:n_samples/2]
Now predict the value of the digit on the second half
data_test, target_test = data[n_samples/2:], target[n_samples/2:]
digit_target_test = target[n_samples/2:]
data_test = scaler.transform(data_test)
Create a classifier a support vector classifier
kernel_svm = svm.SVC(gamma=2)
linear_svm = svm.LinearSVC
create pipeline from kernel approximation
and linear svm
feature_map_fourier = RBFSampler(gamma=2, random_state=1)
feature_map_nystroem = Nystroem(gamma=2, random_state=1)
fourier_approx_svm = pipeline.Pipeline([('feature_map', feature_map_fourier),
svm.LinearSVC])
nystroem_approx_svm = pipeline.Pipeline([('feature_map', feature_map_nystroem),
svm.LinearSVC])
fit and predict using linear and kernel svm
kernel_svm_time = time()
kernel_svm.fit(data_train, target_train)
kernel_svm.score(kernel_svm.score(data_test, target_test))
kernel_svm_time = time()
linear_svm_time = time()
51 Miscellaneous examples 713
```

scikitlearn user guide Release 0213
linearsvmfitdatatrain targetstrain
linearsvm score linearsvm scoredatatest targetstest
linearsvm time time linearsvm time
samplesizes 30 nparange1 10
fourierscores
nystroem scores
fouriertimes
nystroem times
forDinsamplesizes
fourierapproxsvmsetparamsfeaturemapncomponentsD
nystroemapproxsvmsetparamsfeaturemapncomponentsD
start time
nystroemapproxsvmfitdatatrain targetstrain
nystroem times appendtime start
start time
fourierapproxsvmfitdatatrain targetstrain
fouriertimes appendtime start
fourierscore fourierapproxsvm scoredatatest targetstest
nystroem score nystroemapproxsvm scoredatatest targetstest
nystroem scores appendnystroem score
fourierscores appendfourierscore
plot the results
pltfigurefigsize8 8
accuracy pltsubplot211
second y axis for timeings
timescale pltsubplot212
accuracyplotsamplesizes nystroem scores labelNystroem approx kernel
timescaleplotsamplesizes nystroem times
labelNystroem approx kernel
accuracyplotsamplesizes fourierscores labelFourier approx kernel
timescaleplotsamplesizes fouriertimes
labelFourier approx kernel
horizontal lines for exact rbf and linear kernels
accuracyplotsamplesizes0 samplesizes1
linearsvm score linearsvm score labellinear svm
timescaleplotsamplesizes0 samplesizes1
linearsvm time linearsvm time labellinear svm
accuracyplotsamplesizes0 samplesizes1
kernelsvm score kernelsvm score labelrbf svm
timescaleplotsamplesizes0 samplesizes1
kernelsvm time kernelsvm time labelrbf svm
vertical line for dataset dimensionality 64
accuracyplot64 64 07 1 labelInfeatures
legends and labels
accuracysettitleClassification accuracy
timescalesettitleTraining times
accuracysetxlimsamplesizes0 samplesizes1
714 Chapter 5 Examples


```
scikitlearn user guide Release 0213
accuracysetxticks
accuracysetylimnmpminfourierscores 1
timescalesetxlabelSampling steps transformed feature dimension
accuracysetylabelClassification accuracy
timescalesetylabelTraining time in seconds
accuracylegendlocbest
timescalelegendlocbest
visualize the decision surface projected down to the first
two principal components of the dataset
pca PCAncomponents8fitdatatrain
X pcatransformdatatrain
Generate grid along first two principal components
multiples nparange2 2 01
steps along first component
first multiples npnewaxis pcacomponents0
steps along second component
second multiples npnewaxis pcacomponents1
combine
grid firstnpnewaxis second npnewaxis
flatgrid gridreshape1 datashape1
title for the plots
titles SVC (Support Vector Classification) with rbf kernel
SVC (Support Vector Classification) linear kernel nwith Fourier rbf feature map n
ncomponents100
SVC (Support Vector Classification) linear kernel nwith Nystroem rbf feature map n
ncomponents100
plttightlayout
pltfigurefigsize12 5
predict and plot
fori clfinenumeratekernelsvm nystroemapproxsvm
fourierapproxsvm
Plot the decision boundary For that we will assign a color to each
point in the mesh xmin xmaxxymin ymax
pltsubplot1 3 i 1
Z clfpredictflatgrid
Put the result into a color plot
Z Zreshapegridshape1
pltcontourfmultiples multiples Z cmappltcmPaired
pltaxisoff
Plot also the training points
pltscatterX 0 X 1 ctargetstrain cmappltcmPaired
edgecolors0 0 0
plttitletitlesi
plttightlayout
pltshow
Total running time of the script 0 minutes 2188 seconds
51 Miscellaneous examples 715
```

scikitlearn user guide Release 0213

52 Examples based on real world datasets

Applications to real world problems with some medium sized datasets or interactive user interface

Note Click here to download the full example code

521 Outlier detection on a real data set

This example illustrates the need for robust covariance estimation on a real data set It is useful both for outlier detection and for a better understanding of the data structure

We selected two sets of two variables from the Boston housing data set as an illustration of what kind of analysis can be done with several outlier detection tools For the purpose of visualization we are working with twodimensional examples but one should be aware that things are not so trivial in highdimension as it will be pointed out

In both examples below the main result is that the empirical covariance estimate as a nonrobust one is highly influenced by the heterogeneous structure of the observations Although the robust covariance estimate is able to focus on the main mode of the data distribution it sticks to the assumption that the data should be Gaussian distributed yielding some biased estimation of the data structure but yet accurate to some extent The OneClass SVM (Support Vector Machine) does not assume any parametric form of the data distribution and can therefore model the complex shape of the data much better

First example

The first example illustrates how robust covariance estimation can help concentrating on a relevant cluster when another one exists Here many observations are confounded into one and break down the empirical covariance estimation Of course some screening tools would have pointed out the presence of two clusters Support Vector Machines Gaussian Mixture Models univariate outlier detection But had it been a highdimensional example none of these could be applied that easily

Second example

The second example shows the ability of the Minimum Covariance Determinant robust estimator of covariance to concentrate on the main mode of the data distribution the location seems to be well estimated although the covariance is hard to estimate due to the bananashaped distribution Anyway we can get rid of some outlying observations The OneClass SVM (Support Vector Machine) is able to capture the real data structure but the difficulty is to adjust its kernel bandwidth parameter so as to obtain a good compromise between the shape of the data scatter matrix and the risk of overfitting the data

716 Chapter 5 Examples

scikitlearn user guide Release 0213

- 52 Examples based on real world datasets 717

scikitlearn user guide Release 0213

```
•
printdoc
Author Virgile Fritsch virgilefritschinriafr
License BSD (Binary Search Decision) 3 clause
import numpy as np
from sklearncovariance import EllipticEnvelope
from sklearnsvm import OneClassSVM
import matplotlib.pyplot as plt
import matplotlibfontmanager
from sklearndatasets import loadboston
Get data
X1 loadbostondata 8 10 two clusters
X2 loadbostondata 5 12 bananashaped
Define classifiers to be used
classifiers
Empirical Covariance EllipticEnvelopesupportfraction1
contamination0261
Robust Covariance Minimum Covariance Determinant
EllipticEnvelopecontamination0261
OCSVM OneClassSVMnu0261 gamma005
colors m g b
legend1
legend2
718 Chapter 5 Examples
```

scikitlearn user guide Release 0213

```
Learn a frontier for outlier detection with several classifiers
xx1 yy1 npmeshgridnplinspace8 28 500 nplinspace3 40 500
xx2 yy2 npmeshgridnplinspace3 10 500 nplinspace5 45 500
fori clfname clf in enumerate classifiersitems
pltfigure1
clffitX1
Z1 clfdecisionfunctionnpcxx1ravel yy1ravel
Z1 Z1reshapexx1shape
legend1clfname pltcontour
xx1 yy1 Z1 levels0 linewidths2 colorscolorsi
pltfigure2
clffitX2
Z2 clfdecisionfunctionnpcxx2ravel yy2ravel
Z2 Z2reshapexx2shape
legend2clfname pltcontour
xx2 yy2 Z2 levels0 linewidths2 colorscolorsi
legend1valueslist listlegend1values
legend1keyslist listlegend1keys
Plot the results shape of the data points cloud
pltfigure1 two clusters
plttitleOutlier detection on a real data set boston housing
pltscatterX1 0 X1 1 colorblack
bboxargs dictboxstyleround fc08
arrowargs dictarrowstyle
pltannotateseveral confounded points xy24 19
xycoordsdata textcoordsdata
xytext13 10 bboxbboxargs arrowpropsarrowargs
pltxlimxx1min xx1max
pltylimyy1min yy1max
pltlegendlegend1valueslist0collections0
legend1valueslist1collections0
legend1valueslist2collections0
legend1keyslist0 legend1keyslist1 legend1keyslist2
locupper center
propmatplotlibfontmanagerFontPropertiessize12
pltlabelaccessibility to radial highways
pltlabelpupilteacher ratio by town
legend2valueslist listlegend2values
legend2keyslist listlegend2keys
pltfigure2 banana shape
plttitleOutlier detection on a real data set boston housing
pltscatterX2 0 X2 1 colorblack
pltxlimxx2min xx2max
pltylimyy2min yy2max
pltlegendlegend2valueslist0collections0
legend2valueslist1collections0
legend2valueslist2collections0
legend2keyslist0 legend2keyslist1 legend2keyslist2
locupper center
propmatplotlibfontmanagerFontPropertiessize12
pltlabel lower status of the population
pltlabelaverage number of rooms per dwelling
pltshow
52 Examples based on real world datasets 719
```

scikitlearn user guide Release 0213

Total running time of the script 0 minutes 3436 seconds

Note Click here to download the full example code

522 Compressive sensing tomography reconstruction with L1 prior Lasso

This example shows the reconstruction of an image from a set of parallel projections acquired along different angles

Such a dataset is acquired in computed tomography CT

Without any prior information on the sample the number of projections required to reconstruct the image is of the

order of the linear size of the image in pixels For simplicity we consider here a sparse image where only pixels

on the boundary of objects have a nonzero value Such data could correspond for example to a cellular material

Note however that most images are sparse in a different basis such as the Haar wavelets Only 17 projections are

acquired therefore it is necessary to use prior information available on the sample its sparsity this is an example of

compressive sensing

The tomography projection operation is a linear transformation In addition to the datafidelity term corresponding

to a linear regression we penalize the L1 norm of the image to account for its sparsity The resulting optimization

problem is called the Lasso We use the class `sklearnlinearmodelLasso` that uses the coordinate descent

algorithm Importantly this implementation is more computationally efficient on a sparse matrix than the projection

operator used here

The reconstruction with L1 penalization gives a result with zero error all pixels are successfully labeled with 0 or 1

even if noise was added to the projections In comparison an L2 penalization `sklearnlinearmodelRidge`

produces a large number of labeling errors for the pixels Important artifacts are observed on the reconstructed image

contrary to the L1 penalization Note in particular the circular artifact separating the pixels in the corners that have

contributed to fewer projections than the central disk

`printdoc`

Author Emmanuelle Gouillart `emmanuellegouillartnsuporg`

License BSD (Binary Search Decision) 3 clause

`import numpy as np`

`from scipy import sparse`

`from scipy import ndimage`

`from sklearnlinearmodel import Lasso`

720 Chapter 5 Examples

```
scikitlearn user guide Release 0213
from sklearn.linear_model import Ridge
import matplotlib.pyplot as plt
def weights(x, dx=1, orig=0):
    x = np.ravel(x)
    floorx = np.floor(x / orig).astype(np.int64)
    alpha = x / orig - floorx
    dx = dx * alpha
    return np.hstack([floorx, floorx + 1]) * alpha
def generate_center_coordinates(x):
    X, Y = np.mgrid[0:lx, 0:ly].astype(np.float64)
    center = lx / 2
    X = X - center
    Y = Y - center
    return X, Y
def build_projection_operator(x, ndir):
    """Compute the tomography design matrix
    Parameters
```

lx : int
linear size of image array
ndir : int
number of angles at which projections are acquired
Returns

p : sparse matrix of shape (ndir, lx, lx, 2)

```
X, Y = generate_center_coordinates(x)
angles = np.linspace(0, np.pi, ndir, endpoint=False)
data_inds, weights, camera_inds = \
    data_unravel_indices(np.arange(lx, 2), \
    data_unravel_indices, np.hstack([data_unravel_indices, \
    data_unravel_indices]))
for i, angle in enumerate(angles):
    Xrot = np.cos(angle) * X - np.sin(angle) * Y
    inds_w, weights, Xrot, dx, orig, Xmin = \
    mask, np.logical_and(inds_w == 0, inds_w == lx)
    weights = list(weights)
    camera_inds = list(camera_inds)
    data_inds = list(data_unravel_indices)
    proj_operator = sparse_coo_matrix(weights, camera_inds, data_inds)
    return proj_operator
def generate_synthetic_data():
    """Synthetic binary data
    rs = np.random.RandomState(0)
    npts = 36
    x, y = npogrid(0, 0, 1, 1)
    mask_outer = x < 2 & y < 2 & x > 2 & y > 2
    52 Examples based on real world datasets 721
```

scikitlearn user guide Release 0213

mask npzerosl l

points l rsrand2 npts

maskpoints0astypenpint points1astypenpint 1

mask ndimagegaussianfiltermask signal npts

res nplogicalandmask maskmean maskouter

returnnplogicalxorres ndimagebinaryerosionres

Generate synthetic images and projections

l 128

projoperator buildprojectionoperatorl l 7

data generatesyntheticdata

proj projoperator dataravel npnewaxis

proj 015 nprandomrandn projshape

Reconstruction with L2 Ridge penalization

rgrridge Ridgealpha02

rgrridgefitprojoperator projravel

recl2 rgrridgecoefreshapel l

Reconstruction with L1 Lasso penalization

the best value of alpha was determined using cross validation

with LassoCV

rglasso Lassoalpha0001

rglassofitprojoperator projravel

recl1 rglassocoeffreshapel l

pltfigurefigsize8 33

pltsubplot131

pltimshowdata cmappltcmgray interpolationnearest

pltaxisoff

plttitleoriginal image

pltsubplot132

pltimshowrecl2 cmappltcmgray interpolationnearest

plttitleL2 penalization

pltaxisoff

pltsubplot133

pltimshowrecl1 cmappltcmgray interpolationnearest

plttitleL1 penalization

pltaxisoff

pltsubplotsadjustspace001 wspace001 top1 bottom0 left0

right1

pltshow

Total running time of the script 0 minutes 9761 seconds

Note Click here to download the full example code

523 Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation

This is an example of applying sklearndecompositionNMF andsklearndecompositionLatentDirichletAllocation on a corpus of documents and extract additive models of the topic structure

722 Chapter 5 Examples

scikitlearn user guide Release 0213

of the corpus The output is a list of topics each represented as a list of terms weights are not shown

Nonnegative Matrix Factorization is applied with two different objective functions the Frobenius norm and the generalized KullbackLeibler divergence The latter is equivalent to Probabilistic Latent Semantic Indexing

The default parameters nsamples nfeatures ncomponents should make the example runnable in a couple of tens of seconds You can try to increase the dimensions of the problem but be aware that the time complexity is polynomial in NMF (Non-Negative Matrix Factorization) In LDA the time complexity is proportional to nsamples iterations

Out

Loading dataset

done in 7911s

Extracting tfidf features for NMF (Non-Negative Matrix Factorization)

done in 0268s

Extracting tf features for LDA

done in 0254s

Fitting the NMF (Non-Negative Matrix Factorization) model Frobenius norm with tfidf features nsamples2000 and nfeatures1000

done in 0406s

Topics in NMF (Non-Negative Matrix Factorization) model Frobenius norm

Topic 0 just people don think like know time good make way really say right ve want

↪did ll new use years

Topic 1 windows use dos using window program os drivers application help software

↪pc running ms screen files version card code work

Topic 2 god jesus bible faith christian christ christians does heaven sin believe

↪lord life church mary atheism belief human love religion

Topic 3 thanks know does mail advance hi info interested email anybody looking card

↪help like appreciated information send list video need

Topic 4 car cars tires miles 00 new engine insurance price condition oil power

↪speed good 000 brake year models used bought

Topic 5 edu soon com send university internet mit ftp mail cc pub article

↪information hope program mac email home contact blood

Topic 6 file problem files format win sound ftp pub read save site help image

↪available create copy running memory self version

Topic 7 game team games year win play season players nhl runs goal hockey toronto

↪division flyers player defense leafs bad teams

Topic 8 drive drives hard disk floppy software card mac computer power scsi

↪controller apple mb 00 pc rom sale problem internal

Topic 9 key chip clipper keys encryption government public use secure enforcement

↪phone nsa communications law encrypted security clinton used legal standard

Fitting the NMF (Non-Negative Matrix Factorization) model generalized KullbackLeibler divergence with tfidf features

↪nsamples2000 and nfeatures1000

done in 1769s

Topics in NMF (Non-Negative Matrix Factorization) model generalized KullbackLeibler divergence

Topic 0 just people don like did know make really right think say things time look

↪way didn ve course probably good

Topic 1 help thanks windows know hi need using does looking anybody appreciated

↪card mail software use info email ftp available pc

Topic 2 does god believe know mean true christians read point jesus christian

↪church come people fact says religion say agree bible

Topic 3 know thanks mail interested like new just bike email edu advance want

↪contact really list heard com post hear information

Topic 4 10 new 30 12 20 50 11 sale 16 15 time 14 old power ago good 100 great offer

↪cost

52 Examples based on real world datasets 723

scikitlearn user guide Release 0213

Topic 5 number 1993 data subject government new numbers provide information space
↳following com research include large note group major time talk

Topic 6 edu problem file com remember try soon article mike files code program sun
↳free send think cases manager little called

Topic 7 game year team games world fact second case won said win division play best
↳clearly claim allow example used doesn

Topic 8 think don drive hard need bit mac make sure read apple going comes disk
↳computer case pretty drives software ve

Topic 9 good just use like doesn got way don ll going does chip better doing bad
↳key want sure bit car

Fitting LDA models with tf features nsamples2000 and nfeatures1000
done in 3167s

Topics in LDA model

Topic 0 edu com mail send graphics ftp pub available contact university list faq ca
↳information cs 1993 program sun uk mit

Topic 1 don like just know think ve way use right good going make sure ll point got
↳need really time doesn

Topic 2 christian think atheism faith pittsburgh new bible radio games alt lot just
↳religion like book read play time subject believe

Topic 3 drive disk windows thanks use card drives hard version pc software file
↳using scsi help does new dos controller 16

Topic 4 hiv health aids disease april medical care research 1993 light information
↳study national service test led 10 page new drug

Topic 5 god people does just good don jesus say israel way life know true fact time
↳law want believe make think

Topic 6 55 10 11 18 15 team game 19 period play 23 12 13 flyers 20 25 22 17 24 16

Topic 7 car year just cars new engine like bike good oil insurance better tires 000
↳thing speed model brake driving performance

Topic 8 people said did just didn know time like went think children came come don
↳took years say dead told started

Topic 9 key space law government public use encryption earth section security moon
↳probe enforcement keys states lunar military crime surface technology

Author Olivier Grisel oliviergrisenstaorg

Lars Buitinck

ChyiKwei Yau chyikweiyaugmailcom

License BSD (Binary Search Decision) 3 clause

```
from time import time
from sklearn.feature_extraction.text import TfidfVectorizer CountVectorizer
from sklearn.decomposition import NMF (Non-Negative Matrix Factorization) LatentDirichletAllocation
from sklearn.datasets import fetch20newsgroups
nsamples 2000
nfeatures 1000
ncomponents 10
ntopwords 20
def print_top_words(model, feature_names, n_top_words):
```

724 Chapter 5 Examples

```
scikitlearn user guide Release 0213
fortopicidx topic inenumeratemodelcomponents
message Topic d topicidx
message joinfeaturenamesi
foriintopicargsortntopwords 11
printmessage
print
Load the 20 newsgroups dataset and vectorize it We use a few heuristics
to filter out useless terms early on the posts are stripped of headers
footers and quoted replies and common English words words occurring in
only one document or in at least 95 of the documents are removed
printLoading dataset
t0 time
dataset fetch20newsgroupssshuffleTrue randomstate1
removeheaders footers quotes
datasamples datasetdatansamples
printdone in 03fs time t0
Use tfidf features for NMF (Non-Negative Matrix Factorization)
printExtracting tfidf features for NMF (Non-Negative Matrix Factorization)
tfidfvectorizer TfidfVectorizermaxdf095 mindf2
maxfeaturesnfeatures
stopwordseenglish
t0 time
tfidf tfidfvectorizerfittransformdatasamples
printdone in 03fs time t0
Use tf raw term count features for LDA
printExtracting tf features for LDA
tfvectorizer CountVectorizermaxdf095 mindf2
maxfeaturesnfeatures
stopwordseenglish
t0 time
tf tfvectorizerfittransformdatasamples
printdone in 03fs time t0
print
Fit the NMF (Non-Negative Matrix Factorization) model
printFitting the NMF (Non-Negative Matrix Factorization) model Frobenius norm with tfidf features
nsamples dand nfeatures d
nsamples nfeatures
t0 time
nmf NMFncomponentsncomponents randomstate1
alpha1 l1ratio5fittfidf
printdone in 03fs time t0
printnTopics in NMF (Non-Negative Matrix Factorization) model Frobenius norm
tfidffeaturenames tfidfvectorizergetfeaturenames
printtopwordsnmf tfidffeaturenames ntopwords
Fit the NMF (Non-Negative Matrix Factorization) model
printFitting the NMF (Non-Negative Matrix Factorization) model generalized KullbackLeibler divergence with
tfidf features nsamples dand nfeatures d
nsamples nfeatures
t0 time
nmf NMFncomponentsncomponents randomstate1
52 Examples based on real world datasets 725
```

scikitlearn user guide Release 0213
betalosskullbackleibler solvermu maxiter1000 alpha1
l1ratio5fittfidf
printdone in 03fs time t0
printnTopics in NMF (Non-Negative Matrix Factorization) model generalized KullbackLeibler divergence
tfidffeaturenames tfidfvectorizergetfeaturenames
printtopwordsnmf tfidffeaturenames ntopwords
printFitting LDA models with tf features
nsamples dand nfeatures d
nsamples nfeatures
lda LatentDirichletAllocationncomponentsncomponents maxiter5
learningmethodonline
learningoffset50
randomstate0
t0 time
ldafittf
printdone in 03fs time t0
printnTopics in LDA model
tffeaturenames tfvectorizergetfeaturenames
printtopwordsllda tffeaturenames ntopwords
Total running time of the script 0 minutes 13781 seconds
Note Click here to download the full example code
524 Faces recognition example using eigenfaces and SVMs
The dataset used in this example is a preprocessed excerpt of the “Labeled Faces in the Wild” aka LFW
httpviswwwcsumassedulfwlwfwnneledtgz 233MB
Expected results for the top 5 most represented people in the dataset
Ariel Sharon 067 092 077 13
Colin Powell 075 078 076 60
Donald Rumsfeld 078 067 072 27
George W Bush 086 086 086 146
Gerhard Schroeder 076 076 076 25
Hugo Chavez 067 067 067 15
Tony Blair 081 069 075 36
avg total 080 080 080 322
726 Chapter 5 Examples

scikitlearn user guide Release 0213

-

52 Examples based on real world datasets 727

scikitlearn user guide Release 0213

•

Out

Total dataset size

nsamples 1288

nfeatures 1850

nclasses 7

Extracting the top 150 eigenfaces from 966 faces

done in 0118s

Projecting the input data on the eigenfaces orthonormal basis

done in 0005s

Fitting the classifier to the training set

done in 36078s

Best estimator found by grid search

SVCC10000 classweightbalanced gamma0005

Predicting peoples names on the test set

done in 0061s

precision recall f1score support

728 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Ariel Sharon 075 046 057 13
Colin Powell 079 087 083 60
Donald Rumsfeld 094 063 076 27
George W Bush 083 098 090 146
Gerhard Schroeder 091 080 085 25
Hugo Chavez 100 053 070 15
Tony Blair 096 075 084 36
accuracy 085 322
macro avg 088 072 078 322
weighted avg 086 085 084 322
6 2 0 5 0 0 0
1 52 0 7 0 0 0
1 3 17 6 0 0 0
0 3 0 143 0 0 0
0 1 0 3 20 0 1
0 4 0 2 1 8 0
0 1 1 6 1 0 27
from time import time
import logging
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import fetch_lfw_people
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.svm import SVC (Support Vector Classification)
print doc
    Display progress logs on stdout
logging.basicConfig(level=logging.INFO, format='%(asctime)s %(message)s')

Download the data if not already on disk and load it as numpy arrays
lfw_people = fetch_lfw_people(min_faces_per_person=70, resize=0.4)
introspect the images arrays to find the shapes for plotting
n_samples, h, w = lfw_people.images.shape
for machine learning we use the 2 data directly as relative pixel
positions info is ignored by this model
X = lfw_people.data
n_features = X.shape[1]
the label to predict is the id of the person
52 Examples based on real world datasets 729
```

```
scikitlearn user guide Release 0213
y lfwpeopletarget
targetnames lfwpeopletargetnames
nclasses targetnamesshape0
printTotal dataset size
printnsamples d nsamples
printnfeatures d nfeatures
printnclasses d nclasses
```

```
Split into a training set and a test set using a stratified k fold
split into a training and testing set
Xtrain Xtest ytrain ytest traintestsplit
X y testsize025 randomstate42
```

```
Compute a PCA (Principal Component Analysis) eigenfaces on the face dataset treated as unlabeled
dataset unsupervised feature extraction dimensionality reduction
ncomponents 150
printExtracting the top deigenfaces from dfaces
ncomponents Xtrainshape0
t0 time
pca PCAncomponentsncomponents svdsolvverrandomized
whitenTruefitXtrain
printdone in 03fs time t0
eigenfaces pcacomponentsreshapencomponents h w
printProjecting the input data on the eigenfaces orthonormal basis
t0 time
Xtrainpca pcatransformXtrain
Xtestpca pcatransformXtest
printdone in 03fs time t0
```

```
Train a SVM (Support Vector Machine) classification model
printFitting the classifier to the training set
t0 time
paramgrid C 1e3 5e3 1e4 5e4 1e5
gamma 00001 00005 0001 0005 001 01
clf GridSearchCVSVCKernelrbf classweightbalanced
paramgrid cv5 iidFalse
clf clffitXtrainpca ytrain
printdone in 03fs time t0
printBest estimator found by grid search
printclfbestestimator
```

```
Quantitative evaluation of the model quality on the test set
printPredicting peoples names on the test set
730 Chapter 5 Examples
```


scikitlearn user guide Release 0213

```
t0 time
ypred clfpredictXtestpca
printdone in 03fs time t0
printclassificationreportytest ypred targetnamestargetnames
printconfusionmatrixytest ypred labelsrangenclasses
```

```
Qualitative evaluation of the predictions using matplotlib
defplotgalleryimages titles h w nrow3 ncol4
Helper function to plot a gallery of portraits
pltfigurefigsize18 ncol 24 nrow
pltsubplotsadjustbottom0 left01 right99 top90 hspace35
foriinrangennrow ncol
pltsubplotnrow ncol i 1
pltimshowimagesiresshapeh w cmappltcmgray
plttitletitlesi size12
pltxticks
plttyticks
plot the result of the prediction on a portion of the test set
deftitleypred ytest targetnames i
predname targetnamesypredirsplit 11
truenam targetnamesytestirsplit 11
returnpredicted sntrue s predname truenam
predictiontitles titleypred ytest targetnames i
foriinrangeypredshape0
plotgalleryXtest predictiontitles h w
plot the gallery of the most significant eigenfaces
eigenfacetitles eigenface d iforiinrangeeeigenfacesshape0
plotgalleryeigenfaces eigenfacetitles h w
pltshow
```

Total running time of the script 0 minutes 59514 seconds

Note [Click here to download the full example code](#)

525 Model Complexity Influence

Demonstrate how model complexity influences both prediction accuracy and computational performance

The dataset is the Boston Housing dataset resp 20 Newsgroups for regression resp classification

For each class of models we make the model complexity vary through the choice of relevant model parameters and measure the influence on both computational performance latency and predictive power MSE or Hamming Loss

52 Examples based on real world datasets 731

scikitlearn user guide Release 0213

-
-

732 Chapter 5 Examples

scikitlearn user guide Release 0213

- Out
Benchmarking SGDClassifier(alpha=0.001, l1_ratio=0.25, loss=modifiedhuber, penalty=elasticnet)
Complexity 4466 Hamming Loss Misclassification Ratio 0.2491 Pred Time 0.021496s
Benchmarking SGDClassifier(alpha=0.001, l1_ratio=0.05, loss=modifiedhuber, penalty=elasticnet)
Complexity 1663 Hamming Loss Misclassification Ratio 0.2915 Pred Time 0.017370s
Benchmarking SGDClassifier(alpha=0.001, l1_ratio=0.75, loss=modifiedhuber, penalty=elasticnet)
Complexity 880 Hamming Loss Misclassification Ratio 0.3180 Pred Time 0.012989s
Benchmarking SGDClassifier(alpha=0.001, l1_ratio=0.9, loss=modifiedhuber, penalty=elasticnet)
Complexity 639 Hamming Loss Misclassification Ratio 0.3337 Pred Time 0.011292s
Benchmarking NuSVRC(10000, gamma=30517578125e05, nu=0.1)
Complexity 69 MSE 318139 Pred Time 0.000283s
Benchmarking NuSVRC(10000, gamma=30517578125e05, nu=0.25)
Complexity 136 MSE 256140 Pred Time 0.000506s
Benchmarking NuSVRC(10000, gamma=30517578125e05)
Complexity 244 MSE 223375 Pred Time 0.000868s
Benchmarking NuSVRC(10000, gamma=30517578125e05, nu=0.75)
Complexity 351 MSE 213688 Pred Time 0.001226s
Benchmarking NuSVRC(10000, gamma=30517578125e05, nu=0.9)
Complexity 404 MSE 211033 Pred Time 0.001402s
Benchmarking GradientBoostingRegressor(n_estimators=10)
52 Examples based on real world datasets 733

```
scikitlearn user guide Release 0213
Complexity 10 MSE 290148 Pred Time 0000093s
Benchmarking GradientBoostingRegressornestimators50
Complexity 50 MSE 89630 Pred Time 0000165s
Benchmarking GradientBoostingRegressor
Complexity 100 MSE 77187 Pred Time 0000227s
Benchmarking GradientBoostingRegressornestimators200
Complexity 200 MSE 66955 Pred Time 0000608s
Benchmarking GradientBoostingRegressornestimators500
Complexity 500 MSE 71437 Pred Time 0000776s
printdoc
  Author Eustache Diemert eustachediemertfr
  License BSD (Binary Search Decision) 3 clause
import time
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.axesgrid1parasiteaxes import hostssubplot
from mpl_toolkits.axisartistaxislines import Axes
from scipy.sparse import csr_matrix
from sklearn import datasets
from sklearn.utils import shuffle
from sklearn.metrics import mean_squared_error
from sklearn.svm.classes import NuSVR
from sklearn.ensemble.gradientboosting import GradientBoostingRegressor
from sklearn.linear_model.stochastic_gradient import SGDClassifier
from sklearn.metrics import hamming_loss
```

```
Routines
Initialize random generator
np.random.seed(0)
def generate_data(case, sparse=False):
    Generate regression/classification data
    bunch = None
    if case == 'regression':
        bunch = datasets.load_boston
    elif case == 'classification':
        bunch = datasets.fetch_20newsgroups_vectorized(subset='all')
    X, y = shuffle(bunch.data, bunch.target)
    offset = int(X.shape[0] * 0.8)
    X_train, y_train, X_offset, y_offset = \
        X[:offset], y[:offset], X[offset:], y[offset:]
    X_test, y_test, X_offset, y_offset = \
        X[offset:], y[offset:], X[:offset], y[:offset]
734 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
ifsparse
Xtrain csrmatrixXtrain
Xtest csrmatrixXtest
else
Xtrain nparrayXtrain
Xtest nparrayXtest
ytest nparrayytest
ytrain nparrayytrain
data Xtrain Xtrain Xtest Xtest ytrain ytrain
ytest ytest
returndata
defbenchmarkinfluenceconf
```

Benchmark influence of changingparam on both MSE and latency

```
predictiontimes
predictionpowers
complexities
forparamvalue inconfchangingparamvalues
conftunedparamsconfchangingparam paramvalue
estimator confestimator conftunedparams
printBenchmarking s estimator
estimatorfitconfdataXtrain confdataytrain
confpostfithookestimator
complexity confcomplexitycomputerestimator
complexitiesappendcomplexity
starttime timetime
forinrangeconfnsamples
ypred estimatorpredictconfdataXtest
elapsedtime timetime starttime floatconfnsamples
predictiontimesappendelapsedtime
predscore confpredictionperformancecomputer
confdataytest ypred
predictionpowersappendpredscore
printComplexity ds4f Pred Time fsn
complexity confpredictionperformancelabel predscores
elapsedtime
returnpredictionpowers predictiontimes complexities
defplotinfluenceconf msevalues predictiontimes complexities
```

Plot influence of model complexity on both accuracy and latency

```
pltfigurefigsize12 6
host hostsubplot111 axesclassAxes
pltsubplotsadjustright075
par1 hosttwinx
hostsetxlabelModel Complexity s confcomplexitylabel
y1label confpredictionperformancelabel
y2label Time s
hostsetylabely1label
par1setylabely2label
p1 hostplotcomplexities msevalues b labelprediction error
p2 par1plotcomplexities predictiontimes r
labellatency
52 Examples based on real world datasets 735
```

```
scikitlearn user guide Release 0213
hostlegendlocupper right
hostaxisleftlabelsetcolorp1getcolor
par1axisrightlabelsetcolorp2getcolor
plttitleInfluence of Model Complexity s confestimatorname
pltshow
defcountnonzerocoefficientsestimator
a_estimatorcoeftoarray
returnnpcountnonzeroa
```

```
Main code
regressiondata generatedataregression
classificationdata generatedataclassification sparseTrue
configurations
estimator SGDClassifier
tunedparams penalty elasticnet alpha 0001 loss
modifiedhuber fitintercept True tol 1e3
changingparam l1ratio
changingparamvalues 025 05 075 09
complexitylabel nonzero coefficients
complexitycomputer countnonzerocoefficients
predictionperformancecomputer hammingloss
predictionperformancelabel Hamming Loss Misclassification Ratio
postfithook lambdax xsparsify
data classificationdata
nsamples 30
estimator NuSVR
tunedparams C 1e3 gamma 2 15
changingparam nu
changingparamvalues 01 025 05 075 09
complexitylabel nsupportvectors
complexitycomputer lambdax lenxsupportvectors
data regressiondata
postfithook lambdax x
predictionperformancecomputer meansquarederror
predictionperformancelabel MSE
nsamples 30
estimator GradientBoostingRegressor
tunedparams loss ls
changingparam nestimators
changingparamvalues 10 50 100 200 500
complexitylabel ntrees
complexitycomputer lambdax xnestimators
data regressiondata
postfithook lambdax x
predictionperformancecomputer meansquarederror
predictionperformancelabel MSE
nsamples 30
```

```
forconfinconfigurations
predictionperformances predictiontimes complexities
benchmarkinfluenceconf
plotinfluenceconf predictionperformances predictiontimes
complexities
Total running time of the script 0 minutes 35625 seconds
736 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Note Click here to download the full example code

526 Species distribution modeling

Modeling species’ geographic distributions is an important problem in conservation biology In this example we model the geographic distribution of two south american mammals given past observations and 14 environmental variables Since we have only positive examples there are no unsuccessful observations we cast this problem as a density estimation problem and use the OneClassSVM provided by the package sklearnsvm as our modeling tool The dataset is provided by Phillips et al 2006 If available the example uses basemap to plot the coast lines and national boundaries of South America

The two species are

- “Bradypus variegatus” the Browthroated Sloth
- “Microryzomys minutus” also known as the Forest Small Rice Rat a rodent that lives in Peru Colombia Ecuador Peru and Venezuela

References

- “Maximum entropy modeling of species geographic distributions” S J Phillips R P Anderson R E Schapire Ecological Modelling 190231259 2006

52 Examples based on real world datasets 737

scikitlearn user guide Release 0213
Out

Modeling distribution of species bradypus variegatus
fit OneClassSVM done
plot coastlines from coverage
predict species distribution
Area under the ROC curve 0868443

Modeling distribution of species microryzomys minutus
fit OneClassSVM done
plot coastlines from coverage
predict species distribution
Area under the ROC curve 0993919
time elapsed 2018s
Authors Peter Prettenhofer peterprettenhofergmailcom
Jake Vanderplas vanderplasastrowashingtonedu

License BSD (Binary Search Decision) 3 clause
from time import time
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets.base import Bunch
from sklearn.datasets import fetch_species_distributions
from sklearn.datasets.species_distributions import construct_grids
from sklearn import svm metrics
if basemap is available well use it
otherwise well improvise later
try
from mpl_toolkits.basemap import Basemap
basemap True
except ImportError
basemap False
print doc
def create_species_bunch(species_name, train_test_coverages, xgrid, ygrid)
Create a bunch with information about a particular organism
This will use the test/train record arrays to extract the
data specific to the given species name

bunch = Bunch(name=species_name, split2=species_name.encode('ascii')
738 Chapter 5 Examples


```
scikitlearn user guide Release 0213
points dicttesttest traintrain
forlabel pts inpointsitems
    choose points associated with the desired species
pts ptsptsspecies speciesname
bunchpts s label pts
    determine coverage values for each of the training testing points
ix npsearchsortedxgrid ptsdd long
iy npsearchsortedygrid ptsdd lat
bunchcov s label coverages iy ixT
returnbunch
defplotspeciesdistributionspeciesbradypusvariegatus0
microryzomysminutus0
```

Plot the species distribution

```
iflenspecies 2
printNote when more than two species are provided
    only the first two will be used
t0 time
    Load the compressed data
data fetchspeciesdistributions
    Set up the data grid
xgrid ygrid constructgridsdata
    The grid in xy coordinates
X Y npmeshgridxgrid ygrid1
    create a bunch for each species
BVbunch createspeciesbunchspecies0
datatrain datatest
datacoverages xgrid ygrid
MMbunch createspeciesbunchspecies1
datatrain datatest
datacoverages xgrid ygrid
    background points grid coordinates for evaluation
nprandomseed13
backgroundpoints npcnpandomrandintlow0 highdataNy
size10000
nprandomrandintlow0 highdataNx
size10000T
    Well make use of the fact that coverages6 has measurements at all
    land points This will help us decide between land and water
landreference datacoverages6
    Fit predict and plot for each species
fori species inenumerateBVbunch MMbunch
print80
printModeling distribution of species s speciesname
52 Examples based on real world datasets 739
```

```
scikitlearn user guide Release 0213
Standardize features
mean speciescovtrainmeanaxis0
std speciescovtrainstdaxis0
traincoverstd speciescovtrain mean std
Fit OneClassSVM
print fit OneClassSVM end
clf svmOneClassSVMnu01 kernelrbf gamma05
clffittraincoverstd
printdone
Plot map of South America
pltsubplot1 2 i 1
ifbasemap
print plot coastlines using basemap
m Basemapprojectioncyl llcrnrlatYmin
urcrnrlatYmax llcrnrlonXmin
urcrnrlonXmax resolutionc
mdrawcoastlines
mdrawcountries
else
print plot coastlines from coverage
pltcontourX Y landreference
levels9998 colorsk
linestylelessolid
pltxticks
pltyticks
print predict species distribution
Predict species distribution using the training data
Z nponesdataNy dataNx dtypefloat64
Well predict only for the land points
idx npwherelandreference 9999
coveragesland datacoverages idx0 idx1T
pred clfdecisionfunctioncoveragesland mean std
Z predmin
Zidx0 idx1 pred
levels nplinspaceZmin Zmax 25
Zlandreference 9999 9999
plot contours of the prediction
pltcontourfX Y Z levelslevels cmappltcmReds
pltcolorbarformat 2f
scatter trainingtesting points
pltscatterspeciesptstraindd long speciesptstraindd lat
s22 cblack
marker labeltrain
pltscatterspeciesptstestdd long speciesptstestdd lat
s22 cblack
markerx labeltest
pltlegend
plttitlespeciesname
740 Chapter 5 Examples
```

scikitlearn user guide Release 0213

pltaxisequal

Compute AUC with regards to background points

predbackground Zbackgroundpoints0 backgroundpoints1

predtest clfdecisionfunctionspeciescovtest mean std

scores nprpredtest predbackground

y nprnpnespredtestshape npzerospredbackgroundshape

fpr tpr thresholds metricsroccurve scores

rocauc metricsaucfpr tpr

plttext35 70 AUC 3f rocauc haright

printnArea under the ROC curve f rocauc

printntime elapsed 2fs time t0

plotspeciesdistribution

pltshow

Total running time of the script 0 minutes 20185 seconds

Note Click here to download the full example code

527 Visualizing the stock market structure

This example employs several unsupervised learning techniques to extract the stock market structure from variations in historical quotes

The quantity that we use is the daily variation in quote price quotes that are linked tend to cofluctuate during a day

Learning a graph structure

We use sparse inverse covariance estimation to find which quotes are correlated conditionally on the others Specifi cally sparse inverse covariance gives us a graph that is a list of connection For each symbol the symbols that it is connected too are those useful to explain its fluctuations

Clustering

We use clustering to group together quotes that behave similarly Here amongst the various clustering techniques available in the scikitlearn we use Affinity Propagation as it does not enforce equalsize clusters and it can choose automatically the number of clusters from the data

Note that this gives us a different indication than the graph as the graph reflects conditional relations between variables while the clustering reflects marginal properties variables clustered together can be considered as having a similar impact at the level of the full stock market

Embedding in 2D space

For visualization purposes we need to lay out the different symbols on a 2D canvas For this we use Manifold learning techniques to retrieve 2D embedding

52 Examples based on real world datasets 741

scikitlearn user guide Release 0213

Visualization

The output of the 3 models are combined in a 2D graph where nodes represents the stocks and edges the

- cluster labels are used to define the color of the nodes
- the sparse covariance model is used to display the strength of the edges
- the 2D embedding is used to position the nodes in the plan

This example has a fair amount of visualizationrelated code as visualization is crucial here to display the graph One of the challenge is to position the labels minimizing overlap For this we use an heuristic based on the direction of the nearest neighbor along each axis

Out

Cluster 1 Apple Amazon Yahoo

Cluster 2 Comcast Cablevision Time Warner

Cluster 3 ConocoPhillips Chevron Total Valero Energy Exxon

Cluster 4 Cisco Dell HP IBM Microsoft SAP Texas Instruments

Cluster 5 Boeing General Dynamics Northrop Grumman Raytheon

Cluster 6 AIG American express Bank of America Caterpillar CVS DuPont de

↳Nemours Ford General Electrics Goldman Sachs Home Depot JPMorgan Chase

↳Marriott 3M Ryder Wells Fargo WalMart

Cluster 7 McDonalds

742 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Cluster 8 GlaxoSmithKline Novartis Pfizer SanofiAventis Unilever
Cluster 9 Kellogg Coca Cola Pepsi
Cluster 10 ColgatePalmolive KimberlyClark Procter Gamble
Cluster 11 Canon Honda Navistar Sony Toyota Xerox
Author Gael Varoquaux gaelvaroquauxnormalesuporg
License BSD (Binary Search Decision) 3 clause
import sys
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.collections import LineCollection
import pandas as pd
from sklearn import cluster covariance manifold
printdoc
```

Retrieve the data from Internet
The data is from 2003 2008 This is reasonably calm not too long ago so
that we get hightech firms and before the 2008 crash This kind of
historical data can be obtained for from APIs like the quandlcom and
alphavantageco ones
symboldict
TOT Total
XOM Exxon
CVX Chevron
COP ConocoPhillips
VLO Valero Energy
MSFT Microsoft
IBM IBM
TWX Time Warner
CMCSA Comcast
CVC Cablevision
YHOO Yahoo
DELL Dell
HPQ HP
AMZN Amazon
TM Toyota
CAJ Canon
SNE Sony
F Ford
HMC Honda
NAV Navistar
NOC Northrop Grumman
BA Boeing
KO Coca Cola
52 Examples based on real world datasets 743

scikitlearn user guide Release 0213

MMM 3M
MCD McDonald s
PEP Pepsi
K Kellogg
UN Unilever
MAR Marriott
PG Procter Gamble
CL ColgatePalmolive
GE General Electrics
WFC Wells Fargo
JPM JPMorgan Chase
AIG AIG
AXP American express
BAC Bank of America
GS Goldman Sachs
AAPL Apple
SAP SAP
CSCO Cisco
TXN Texas Instruments
XRX Xerox
WMT WalMart
HD Home Depot
GSK GlaxoSmithKline
PFE Pfizer
SNY SanofiAventis
NVS Novartis
KMB KimberlyClark
R Ryder
GD General Dynamics
RTN Raytheon
CVS CVS
CAT Caterpillar
DD DuPont de Nemours
symbols names nparraysortedsymboldictitemsT
quotes
forsymbolinsymbols
printFetching quote history for r symbol filesysstderr
url httpsrawgithubusercontentcomscikitlearnexamplesdata
masterfinancialdatacsv
quotesappendpdreadcsvurlformatsymbol
closeprices npvstackqclose forqinquotes
openprices npvstackqopen forqinquotes
The daily variations of the quotes are what carry most information
variation closeprices openprices

Learn a graphical structure from the correlations
edgemodel covarianceGraphicalLassoCVcv5
standardize the time series using correlations rather than covariance
is more efficient for structure recovery
744 Chapter 5 Examples

scikitlearn user guide Release 0213

X variationcopyT
X Xstdaxis0
edgemodelfitX

Cluster using affinity propagation
labels clusteraffinitypropagationedgemodelcovariance
nlabels labelsmax
foriinrangenlabels 1
printCluster is i 1 joinnameslabels i

Find a lowdimension embedding for visualization find the best position of
the nodes the stocks on a 2D plane
We use a dense eigensolver to achieve reproducibility arpack is
initiated with random vectors that we dont control In addition we
use a large number of neighbors to capture the largescale structure
nodepositionmodel manifoldLocallyLinearEmbedding
ncomponents2 eigensolverdense nneighbors6
embedding nodepositionmodelfittransformXTT

Visualization
pltfigure1 facecolorw figsize10 8
pltclf
ax pltaxes0 0 1 1
pltaxisoff
Display a graph of the partial correlations
partialcorrelations edgemodelprecisioncopy
d 1 npsqrtnpdiaipartialcorrelations
partialcorrelations d
partialcorrelations d npnewaxis
nonzero npabsnptriupartialcorrelations k1 002
Plot the nodes using the coordinates of our embedding
pltscatterembedding0 embedding1 s100 d2 clabels
cmappltcmnipy spectral
Plot the edges
startidx endidx npwherenonzero
a sequence of line0line1line2 where
linen x0 y0 x1 y1 xm ym
segments embedding start embedding stop
forstart stop inzipstartidx endidx
values npabspartialcorrelationsnonzero
lc LineCollectionsegments
zorder0 cmappltcmhotr
normpltNormalize0 7 valuesmax
lcsetarrayvalues
lcsetlinewidths15 values
axaddcollectionlc
Add a label to each node The challenge here is that we want to
52 Examples based on real world datasets 745

scikitlearn user guide Release 0213
position the labels to avoid overlap with other labels
forindex name label x y inenumerate
zipnames labels embeddingT
dx x embedding0
dxindex 1
dy y embedding1
dyindex 1
thisdx dxnpargminnpabsdy
thisdy dynpargminnpabsdx
ifthisdx 0
horizontalalignment left
x x 002
else
horizontalalignment right
x x 002
ifthisdy 0
verticalalignment bottom
y y 002
else
verticalalignment top
y y 002
plttextx y name size10
horizontalalignmenthorizontalalignment
verticalalignmentverticalalignment
bboxdictfacecolorw
edgecolorpltcmnipy spectrallabel floatnlabels
alpha6
pltxlimembedding0min 15 embedding0ptp
embedding0max 10 embedding0ptp
pltylimembedding1min 03 embedding1ptp
embedding1max 03 embedding1ptp
pltshow
Total running time of the script 0 minutes 4857 seconds
Note Click here to download the full example code
528 Wikipedia principal eigenvector
A classical way to assert the relative importance of vertices in a graph is to compute the principal eigenvector of the adjacency matrix so as to assign to each vertex the values of the components of the first eigenvector as a centrality score
https://en.wikipedia.org/wiki/Eigenvector_centrality
On the graph of webpages and links those values are called the PageRank scores by Google
The goal of this example is to analyze the graph of links inside wikipedia articles to rank articles by relative importance according to this eigenvector centrality
The traditional way to compute the principal eigenvector is to use the power iteration method
https://en.wikipedia.org/wiki/Power_iteration
746 Chapter 5 Examples

scikitlearn user guide Release 0213

Here the computation is achieved thanks to Martinsson’s Randomized SVD algorithm implemented in scikitlearn
The graph data is fetched from the DBpedia dumps DBpedia is an extraction of the latent structured data of the Wikipedia content

```
Author Olivier Grisel oliviergrisenstaorg
License BSD (Binary Search Decision) 3 clause
from bz2 import BZ2File
import os
from datetime import datetime
from pprint import pprint
from time import time
import numpy as np
from scipy import sparse
from joblib import Memory
from sklearn.decomposition import randomizedsvd
from urllib.request import urlopen
printdoc
```

```
Where to download the data if not already on disk
redirectsurl httpdownloadsdwpediaorg351enredirectsenntbz2
redirectsfilename redirectsurlsplit 11
pagelinksurl httpdownloadsdwpediaorg351enpagelinksenntbz2
pagelinksfilename pagelinksurlsplit 11
resources
redirectsurl redirectsfilename
pagelinksurl pagelinksfilename
```

```
forurl filename inresources
if notospathexistsfilename
printDownloading data from s please wait url
opener urlopenurl
openfilename wbwriteopenerread
print
```

```
Loading the redirect files
memory Memorycachedir
defindexredirects indexmap k
Find the index of an article name after redirect resolution
k redirectsgetk k
returnindexmapsetdefaultk lenindexmap
52 Examples based on real world datasets 747
```

```
scikitlearn user guide Release 0213
DBPEDIARESOURCEPREFIXLEN lenhttpdbpediaorgresource
SHORTNAMESLICE sliceDBPEDIARESOURCEPREFIXLEN 1 1
defshortnamenturi
Remove the and URI markers and the common URI prefix
returnnturiSHORTNAMESLICE
defgetredirectsredirectsfilename
Parse the redirections and build a transitively closed map out of it
redirects
printParsing the NT redirect file
forl line inenumerateBZ2Filerredirectsfilename
split linesplit
iflensplit 4
printignoring malformed line line
continue
redirectsshortnamesplit0 shortnamesplit2
ifl 1000000 0
prints line 08d datetimenowisoformat l
compute the transitive closure
printComputing the transitive closure of the redirect relation
forl source inenumeraterredirectskeys
transitivetarget None
target redirectssource
seen source
whileTrue
transitivetarget target
target redirectsgettarget
iftargetisNoneortargetinseen
break
seenaddtarget
redirectssource transitivetarget
ifl 1000000 0
prints line 08d datetimenowisoformat l
returnredirects
disabling joblib as the pickling of large dicts seems much too slow
memorycache
defgetadjacencymatrixredirectsfilename pagelinksfilename limitNone
Extract the adjacency graph as a scipy sparse matrix
Redirects are resolved first
Returns X the scipy sparse adjacency matrix redirects as python
dict from article names to article names and indexmap a python dict
from article names to python int article indexes

printComputing the redirect map
redirects getredirectsredirectsfilename
printComputing the integer index map
indexmap dict
748 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
links list
for l in enumerate(BZ2Filepagelinksfilename
split_linesplit
if len(split) > 4
print ignoring malformed line line
continue
i_indexredirects_indexmap_shortnamesplit0
j_indexredirects_indexmap_shortnamesplit2
linksappend i
if l > 1000000: 0
print line 08d datetimenowisoformat l
if limit is not None and l > limit: 1
break
print Computing the adjacency matrix
X = sparse_lil_matrix(len_indexmap, len_indexmap, dtype=np.float32)
for j in links:
    Xi[j] = 10
del links
print Converting to CSR (Compressed Sparse Row) representation
X = X.tocsr()
print CSR conversion done
return X.redirects_indexmap
# stop after 5M links to make it possible to work in RAM
X.redirects_indexmap.get_adjacency_matrix
redirectsfilename pagelinksfilename limit=5000000
names = [i.name for i in indexmap.items]
print Computing the principal singular vectors using randomizedsvd
t0 = time
U, s, V = randomizedsvd(X, 5, niter=3)
print done in %fs time %t0
print the names of the wikipedia related strongest components of the
principal singular vector which should be similar to the highest eigenvector
print Top wikipedia pages according to principal singular vectors
pprint(names[i] for i in np.argsort(U.T)argsort10)
pprint(names[i] for i in np.argsort(V)argsort10)
def centrality_scores(X, alpha=0.85, max_iter=100, tol=1e-10):
    """Power iteration computation of the principal eigenvector
    This method is also known as Google PageRank and the implementation
    is based on the one from the NetworkX project BSD (Binary Search Decision) licensed too
    with copyrights by
    Aric Hagberg <hagberg@lanl.gov>
    Dan Schult <dschult@colgate.edu>
    Pieter Swart <swart@lanl.gov>"""
    n = X.shape[0]
    X = X.copy()
    incoming_counts = np.asarray(X.sum(axis=1).ravel)
    52 Examples based on real world datasets 749
```

```
scikitlearn user guide Release 0213
printNormalizing the graph
foriinincomingcountsnonzero0
XdataXindptriXindptri 1 10 incomingcountsi
dangle npasarraynpwherenpiscloseXsumaxis1 0
10 n 0ravel
scores npfulln 1 n dtype npfloat32 initial guess
foriinrangemaxiter
printpower iteration d i
prevscores scores
scores alpha scores X npdotdangle prevscores
1 alpha prevscoressum n
check convergence normalized linf norm
scoresmax npabsscoresmax
ifscoresmax 00
scoresmax 10
err npabsscores prevscoresmax scoresmax
printerror 06f err
iferr n tol
returnscores
returnscores
printComputing principal eigenvector score using a power iteration method
t0 time
scores centralityscoresX maxiter100 tol1e10
printdone in 03fs time t0
pprintnamesi foriinnpabsscoresargsort10
Total running time of the script 0 minutes 0000 seconds
Note Click here to download the full example code
529 Libsvm GUI
A simple graphical frontend for Libsvm mainly intended for didactic purposes You can create data points by point
and click and visualize the decision region induced by different kernels and parameter settings
To create positive examples click the left mouse button to create negative examples click the right button
If all examples are from the same class it uses a oneclass SVM (Support Vector Machine)
printdoc
Author Peter Prettenhoer peterprettenhofergmailcom

License BSD (Binary Search Decision) 3 clause
import matplotlib
matplotlibuseTkAgg
from matplotlibbackendsbackendtkagg import FigureCanvasTkAgg
from matplotlibbackendsbackendtkagg import NavigationToolbar2TkAgg
from matplotlibfigure import Figure
from matplotlibcontour import ContourSet
750 Chapter 5 Examples
```

scikitlearn user guide Release 0213

```
import sys
import numpy as np
import tkinter as Tk
from sklearn import svm
from sklearn.datasets import dumpsvmlightfile
ymin ymax 50 50
xmin xmax 50 50
class Model
The Model which hold the data It implements the
observable in the observer pattern and notifies the
registered observers on change event
```

```
definitself
selfobservers
selfsurface None
selfdata
selfcls None
selfsurfacetype 0
defchangedself event
Notify the observers
forobserver inselfobservers
observerupdateevent self
defaddobserverself observer
Register an observer
selfobserversappendobserver
defsetsurface self surface
selfsurface surface
defdumpsvmlightfile self file
data nparrayselfdata
X data 02
y data 2
dumpsvmlightfileX y file
class Controller
definitself model
selfmodel model
selfkernel TkIntVar
selfsurfacetype TkIntVar
Whether or not a model has been fitted
selffitted False
deffitself
printfit the model
train nparrayselfmodeldata
X train 02
y train 2
52 Examples based on real world datasets 751
```

```
scikitlearn user guide Release 0213
C floatselfcomplexityget
gamma floatselfgammaget
coef0 floatselfcoef0get
degree intselfdegreeget
kernelmap 0 linear 1 rbf 2 poly
iflennpuniquey 1
clf svmOneClassSVMkernelkernelmapselfkernelget
gammagamma coef0coef0 degreedegree
clffitX
else
clf svmSVCKernelkernelmapselfkernelget CC
gammagamma coef0coef0 degreedegree
clffitX y
ifhasattrclf score
printAccuracy clfscoreX y 100
X1 X2 Z selfdecisionsurfaceclf
selfmodelclf clf
selfmodelsetsurfaceX1 X2 Z
selfmodelsurfacetype selfsurfacetypeget
selffitted True
selfmodelchangedsurface
defdecisionsurfaceself cls
delta 1
x nparangexmin xmax delta delta
y nparangeymax ymax delta delta
X1 X2 npmeshgridx y
Z clsdecisionfunctionnpcX1ravel X2ravel
Z ZreshapeX1shape
returnX1 X2 Z
defcleardataself
selfmodeldata
selffitted False
selfmodelchangedclear
defaddexampleself x y label
selfmodeldataappendx y label
selfmodelchangedexampleadded
update decision surface if already fitted
selfrefit
defrefitself
Refit the model if already fitted
ifselffitted
selfffit
class View
Test docstring
definitself root controller
f Figure
ax faddsubplot111
axsetxticks
axsetyticks
axsetxlimxmin xmax
axsetylimymin ymax
752 Chapter 5 Examples
```

scikitlearn user guide Release 0213
canvas FigureCanvasTkAgg masterroot
canvasshow
canvasgettkwidgetpacksideTkTOP fillTkBOTH expand1
canvastkcanvaspacksideTkTOP fillTkBOTH expand1
canvasmplconnectbuttononpressevent selfonclick
toolbar NavigationToolbar2TkAgg canvas root
toolbarupdate
selfcontrollbar ControllBarroot controller
selfff f
selfax ax
selfcanvas canvas
selfcontroller controller
selfcontours
selfclabels None
selfplotkernels
defplotkernelself
selfaxtext50 60 Linear uT v
selfaxtext20 60 rRBF exp gamma uv 2
selfaxtext10 60 rPoly gamma uT v rd
defonclickself event
ifeventxdata andeventydata
ifeventbutton 1
selfcontrolleraddexampleeventxdata eventydata 1
elifeventbutton 3
selfcontrolleraddexampleeventxdata eventydata 1
defupdateexampleself model idx
x y l modeldataidx
ifl 1
color w
elifl 1
color k
selfaxplotx y so color scalex00 scaley00
defupdateexampleself event model
ifevent examplesloaded
foriinrangelenmodeldata
selfupdateexamplemodel i
ifevent exampleadded
selfupdateexamplemodel 1
ifevent clear
selfaxclear
selfaxsetxticks
selfaxsetyticks
selfcontours
selfclabels None
selfplotkernels
ifevent surface
selfremovesurface
selfplotsupportvectorsmodelclf supportvectors
selfplotdecisionsurfacemodelsurface modelsurfacetyp
selfcanvasdraw
52 Examples based on real world datasets 753

```
scikitlearn user guide Release 0213
defremovesurfaceself
Remove old decision surface
iflenselfcontours 0
forcontour inselfcontours
ifisinstancecontour ContourSet
forlineset incontourcollections
linesetremove
else
contourremove
selfcontours
defplotsupportvectorsself supportvectors
Plot the support vectors by placing circles over the
corresponding data points and adds the circle collection
to the contours list
cs selfaxscattersupportvectors 0 supportvectors 1
s80 edgecolorsk facecolorsnone
selfcontoursappendcs
defplotdecisionsurfaceself surface type
X1 X2 Z surface
iftype 0
levels 10 00 10
linestyles dashed solid dashed
colors k
selfcontoursappendselfaxcontourX1 X2 Z levels
colorscolors
linestyleslinestyles
eliftype 1
selfcontoursappendselfaxcontourfX1 X2 Z 10
cmapmatplotlibcm_bone
originlower alpha085
selfcontoursappendselfaxcontourX1 X2 Z 00 colorsk
linestylelessolid
else
raiseValueErrorssurface type unknown
class ControllBar
definitself root controller
fm TkFrameroot
kernelgroup TkFramefm
TkRadiobuttonkernelgroup textLinear variablecontrollerkernel
value0 commandcontrollerrefitpackanchorTkW
TkRadiobuttonkernelgroup textRBF variablecontrollerkernel
value1 commandcontrollerrefitpackanchorTkW
TkRadiobuttonkernelgroup textPoly variablecontrollerkernel
value2 commandcontrollerrefitpackanchorTkW
kernelgrouppacksideTkLEFT
valbox TkFramefm
controllercomplexity TkStringVar
controllercomplexityset10
c TkFramevalbox
TkLabelc textC anchore width7packsideTkLEFT
TkEntryc width6 textvariablecontrollercomplexitypack
sideTkLEFT
754 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
cpack
controllergamma TkStringVar
controllergammaset001
g TkFramevalbox
TkLabelg textgamma anchore width7packsideTkLEFT
TkEntryg width6 textvariablecontrollergammapacksideTkLEFT
gpack
controllerdegree TkStringVar
controllerdegreeset3
d TkFramevalbox
TkLabeld textdegree anchore width7packsideTkLEFT
TkEntryd width6 textvariablecontrollerdegreepacksideTkLEFT
dpack
controllercoef0 TkStringVar
controllercoef0set0
r TkFramevalbox
TkLabelr textcoef0 anchore width7packsideTkLEFT
TkEntryr width6 textvariablecontrollercoef0packsideTkLEFT
rpack
valboxpacksideTkLEFT
cmapgroup TkFramefm
TkRadiobuttoncmapgroup textHyperplanes
variablecontrollersurfacetype value0
commandcontrollerrefitpackanchorTkW
TkRadiobuttoncmapgroup textSurface
variablecontrollersurfacetype value1
commandcontrollerrefitpackanchorTkW
cmapgrouppacksideTkLEFT
trainbutton TkButtonfm textFit width5
commandcontrollerfit
trainbuttonpack
fmpacksideTkLEFT
TkButtonfm textClear width5
commandcontrollercleardatapacksideTkLEFT
defgetparser
from optparse import OptionParser
op OptionParser
opaddoptionoutput
actionstore typestr destoutput
helpPath where to dump data
returnop
defmainargv
op getparser
opts args opparseargsargv1
root TkTk
model Model
controller Controllermodel
rootwmtitleScikitlearn Libsvm GUI
52 Examples based on real world datasets 755
```

scikitlearn user guide Release 0213

view Viewroot controller

modeladdobserverview

Tkmainloop

ifoptsoutput

modeldumpsvmlightfileoptsoutput

ifname main

mainsysargv

Total running time of the script 0 minutes 0000 seconds

Note Click here to download the full example code

5210 Prediction Latency

This is an example showing the prediction latency of various scikitlearn estimators

The goal is to measure the latency one can expect when doing predictions either in bulk or atomic ie one by one

mode

The plots represent the distribution of the prediction latency as a boxplot

•

756 Chapter 5 Examples

scikitlearn user guide Release 0213

-
-

52 Examples based on real world datasets 757

scikitlearn user guide Release 0213

```
•
Out
Benchmarking SGDRegressor(alpha=0.01, l1_ratio=0.25, penalty='elasticnet', tol=0.0001)
Benchmarking RandomForestRegressor(n_estimators=100)
Benchmarking SVR (Support Vector Regression)
benchmarking with 100 features
benchmarking with 250 features
benchmarking with 500 features
example run in 960s
Authors Eustache Diemert eustachediemertfr
License BSD (Binary Search Decision) 3 clause
from collections import defaultdict
import time
import gc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.datasets.samples_generator import make_regression
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import Ridge
from sklearn.linear_model import SGDRegressor
from sklearn.svm import SVR (Support Vector Regression)
from sklearn.utils import shuffle
758 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
defnotinsphinx
    Hack to detect whether we are running by the sphinx builder
returnfile inglobals
defatomicbenchmarkestimator(estimator Xtest verbose=False
    Measure runtime prediction of each instance
    ninstances Xtestshape0
    runtimes npzerosninstances dtype=npfloat
    foriinrangeninstances
        instance Xtesti
        start timetime
        estimatorpredictinstance
        runtimesi timetime start
        ifverbose
            printatomicbenchmark runtimes minruntimes nppercentile
            runtimes 50 maxruntimes
            returnruntimes
defbulkbenchmarkestimator(estimator Xtest nbulkrepeats verbose
    Measure runtime prediction of the whole input
    ninstances Xtestshape0
    runtimes npzerosnbulkrepeats dtype=npfloat
    foriinrangenbulkrepeats
        start timetime
        estimatorpredictXtest
        runtimesi timetime start
        runtimes nparraylistmap lambda x floatninstances runtimes
        ifverbose
            printbulkbenchmark runtimes minruntimes nppercentile
            runtimes 50 maxruntimes
            returnruntimes
defbenchmarkestimator(estimator Xtest nbulkrepeats30 verbose=False

    Measure runtimes of prediction in both atomic and bulk mode
    Parameters

    estimator already trained estimator supporting predict
    Xtest test input
    nbulkrepeats how many times to repeat when evaluating bulk mode
    Returns

    atomicruntimes bulkruntimes a pair of nparray which contain the
    runtimes in seconds

    atomicruntimes atomicbenchmarkestimator(estimator Xtest verbose
    bulkruntimes bulkbenchmarkestimator(estimator Xtest nbulkrepeats
    verbose
    returnatomicruntimes bulkruntimes
52 Examples based on real world datasets 759
```

```
scikitlearn user guide Release 0213
defgeneratedatasetntrain ntest nfeatures noise01 verboseFalse
Generate a regression dataset with the given parameters
ifverbose
printgenerating dataset
X y coef makeregressionnsamplesntrain ntest
nfeaturesnfeatures noisenoise coefTrue
randomseed 13
Xtrain Xtest ytrain ytest traintestsplit
X y trainsizentrain testsizentest randomstaterandomseed
Xtrain ytrain shuffleXtrain ytrain randomstaterandomseed
Xscaler StandardScaler
Xtrain XscalerfittransformXtrain
Xtest XscalertransformXtest
yscaler StandardScaler
ytrain yscalerfittransformytrain None 0
ytest yscalertransformytest None 0
gccollect
ifverbose
printok
returnXtrain ytrain Xtest ytest
defboxplotruntimesruntimes predtype configuration

Plot a new Figure with boxplots of prediction runtimes
Parameters

runtimes list of nparray of latencies in microseconds
clsnames list of estimator class names that generated the runtimes
predtype bulk or atomic

fig ax1 pltsubplotsfigsize10 6
bp pltboxplotruntimes
clsinfos snd s estimatorconfname
estimatorconfcomplexitycomputer
estimatorconfinstance
estimatorconfcomplexitylabel for
estimatorconf inconfigurationestimators
pltsetpax1 xticklabelsclsinfos
pltsetpbpboxes colorblack
pltsetpbpwhiskers colorblack
pltsetpbpfliers colored marker
ax1yaxisgridTrue linestyle whichmajor colorlightgrey
alpha05
ax1setaxisbelowTrue
ax1settitlePrediction Time per Instance s dfeats
760 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
predtypecapitalize
configurationnfeatures
ax1setylabelPrediction Time us
pltshow
defbenchmarkconfiguration
Run the whole benchmark
Xtrain ytrain Xtest ytest generatedataset
configurationntrain configurationntest
configurationnfeatures
stats
forestimatorconf inconfigurationestimators
printBenchmarking estimatorconfinstance
estimatorconfinstancefitXtrain ytrain
gccollect
a b benchmarkestimatorconfinstance Xtest
statsestimatorconfname atomic a bulk b
clsnames estimatorconfname forestimatorconf inconfiguration
estimators
runtimes 1e6 statsclfnameatomic forclfname inclsnames
boxplotruntimesruntimes atomic configuration
runtimes 1e6 statsclfnamebulk forclfname inclsnames
boxplotruntimesruntimes bulk d configurationntest
configuration
defnfeatureinfluenceestimators ntrain ntest nfeatures percentile
```

Estimate influence of the number of features on prediction time
Parameters

estimators dict of name str estimator to benchmark
ntrain nber of training instances int
ntest nber of testing instances int
nfeatures list of featurespace dimensionality to test int
percentile percentile at which to measure the speed int 0100
Returns

percentiles dictestimatorname
dictnfeatures percentileperfinus

percentiles defaultdictdefaultdict
forninnfeatures
printbenchmarking with dfeatures n
Xtrain ytrain Xtest ytest generatedatasetntrain ntest n
forclsname estimator inestimatorsitems
estimatorfitXtrain ytrain
gccollect
runtimes bulkbenchmarkestimatorconfinstance Xtest 30 False
52 Examples based on real world datasets 761

```
scikitlearn user guide Release 0213
percentilescslnamen 1e6 nppercentileruntimes
percentile
returnpercentiles
defplotnfeaturesinfluencepercentiles percentile
fig ax1 pltsubplotsfigsize10 6
colors r g b
fori clsname inenumeratepercentileskeys
x nparraysortedn forninpercentilescslnamekeys
y nparraypercentilescslnamen forninx
pltplotx y colorcolorsi
ax1yaxisgridTrue linestyle whichmajor colorlightgrey
alpha05
ax1setaxisbelowTrue
ax1settitleEvolution of Prediction Time with Features
ax1setxlabelFeatures
ax1setylabelPrediction Time at dile us percentile
pltshow
defbenchmarkthroughputsconfiguration durationsecs01
benchmark throughput for different estimators
Xtrain ytrain Xtest ytest generatedataset
configurationntrain configurationntest
configurationnfeatures
throughputs dict
forestimatorconfig inconfigurationestimators
estimatorconfiginstancefitXtrain ytrain
starttime time
npredictions 0
whiletime starttime durationsecs
estimatorconfiginstancepredictXtest0
npredictions 1
throughputsestimatorconfigname npredictions durationsecs
returnthroughputs
defplotbenchmarkthroughputthroughputs configuration
fig ax pltsubplotsfigsize10 6
colors r g b
clsinfos snd s estimatorconfname
estimatorconfcomplexitycomputer
estimatorconfinstance
estimatorconfcomplexitylabel for
estimatorconf inconfigurationestimators
clsvalues throughputsestimatorconfname forestimatorconf in
configurationestimators
pltbarrangelenthroughputs clsvalues width05 colorcolorsi
axsetxticksnplinspace025 lensthroughputs 075 lensthroughputs
axsetxticklabelsclsinfos fontsize10
ymax maxclsvalues 12
axsetylim0 ymax
axsetylabelThroughput predictionssec
axsettitlePrediction Throughput for different estimators d
features configurationnfeatures
pltshow
762 Chapter 5 Examples
```


scikitlearn user guide Release 0213

Main code
starttime timetime

Benchmark bulkatomic prediction speed for various regressors
configuration
ntrain int1e3
ntest int1e2
nfeatures int1e2
estimators
name Linear Model
instance SGDRegressorpenaltyelasticnet alpha001
l1ratio025 fitinterceptTrue
tol1e4
complexitylabel nonzero coefficients
complexitycomputer lambdaclf npcountnonzeroclfcoef
name RandomForest
instance RandomForestRegressornestimators100
complexitylabel estimators
complexitycomputer lambdaclf clfnestimators
name SVR (Support Vector Regression)
instance SVRkernelrbf
complexitylabel support vectors
complexitycomputer lambdaclf lenclfsupportvectors

benchmarkconfiguration
benchmark nfeatures influence on prediction speed
percentile 90
percentiles nfeatureinfluenceridge Ridge
configurationntrain
configurationntest
100 250 500 percentile
plotnfeaturesinfluencepercentiles percentile
benchmark throughput
throughputs benchmarkthroughputsconfiguration
plotbenchmarkthroughputthroughputs configuration
stoptime timetime
printexample run in 2fs stoptime starttime
Total running time of the script 0 minutes 9597 seconds
Note Click here to download the full example code
5211 Outofcore classification of text documents
This is an example showing how scikitlearn can be used for classification using an outofcore approach learning
from data that doesn't fit into main memory We make use of an online classifier ie one that supports the partialfit
method that will be fed with batches of examples To guarantee that the features space remains the same over time
52 Examples based on real world datasets 763

scikitlearn user guide Release 0213

we leverage a HashingVectorizer that will project each example into the same feature space This is especially useful in the case of text classification where new features words may appear in each batch

The dataset used in this example is Reuters21578 as provided by the UCI ML repository It will be automatically downloaded and uncompressed on first run

The plot represents the learning curve of the classifier the evolution of classification accuracy over the course of the minibatches Accuracy is measured on the first 1000 samples held out as a validation set

To limit the memory consumption we queue examples up to a fixed amount before feeding them to the learner

Authors Eustache Diemert eustachediemertfr

FedericoV <https://github.com/FedericoV>

License BSD (Binary Search Decision) 3 clause

from glob import glob

import itertools

import os.path

import re

import tarfile

import time

import sys

import numpy as np

import matplotlib.pyplot as plt

from matplotlib import rcParams

from htmlparser import HTMLParser

from urllibrequest import urlretrieve

from sklearn.datasets import get_data_home

from sklearn.feature_extraction.text import HashingVectorizer

from sklearn.linear_model import SGDClassifier

from sklearn.linear_model import PassiveAggressiveClassifier

from sklearn.linear_model import Perceptron

from sklearn.naive_bayes import MultinomialNB

def not_in_sphinx

 Hack to detect whether we are running by the sphinx builder

return file_in_globals

Reuters Dataset related routines

class ReutersParser(HTMLParser)

 Utility class to parse a SGML file and yield documents one at a time

 def __init__(self, encoding='latin1')

 HTMLParser.__init__(self)

 self.reset

 self.encoding = encoding

 def handle_starttag(self, tag, attrs)

 method start_tag

 getattr(self, method, lambda x: None) if attrs

 def handle_endtag(self, tag)

 method end_tag

 getattr(self, method, lambda: None)

764 Chapter 5 Examples

scikitlearn user guide Release 0213

defresetself

selfintitle 0

selfinbody 0

selfintopics 0

selfintopicd 0

selftitle

selfbody

selftopics

selftopicd

defparseself fd

selfdocs

forchunkinfd

selffeedchunkdecodeselfencoding

fordocinselfdocs

yielddoc

selfdocs

selfclose

defhandledataself data

ifselfinbody

selfbody data

elifselfintitle

selftitle data

elifselfintopicd

selftopicd data

defstartreutersself attributes

pass

defendreutersself

selfbody resubrs r selfbody

selfdocsappendtitle selftitle

body selfbody

topics selftopics

selfreset

defstarttitleself attributes

selfintitle 1

defendtitleself

selfintitle 0

defstartbodyself attributes

selfinbody 1

defendbodyself

selfinbody 0

defstarttopicsself attributes

selfintopics 1

defendtopicsself

selfintopics 0

defstartdself attributes

selfintopicd 1

52 Examples based on real world datasets 765

```
scikitlearn user guide Release 0213
defenddself
selfintopicd 0
selftopicsappendselftopicd
selftopicd
defstreamreutersdocumentsdatapathNone
Iterate over documents of the Reuters dataset
The Reuters archive will automatically be downloaded and uncompressed if
the datapath directory does not exist
Documents are represented as dictionaries with body str
title str topics liststr keys

DOWNLOADURL httparchiveicruciedumlmachinelearningdatabases
reuters21578mldreuters21578targz
ARCHIVEFILENAME reuters21578targz
ifdatapath isNone
datapath ospathjoingetdatahome reuters
if notospathexistsdatapath
Download the dataset
printdownloading dataset once and for all into s
datapath
osmkdirdatapath
defprogressblocknum bs size
totalszmb 2fMB size 1e6
currentszmb 2fMB blocknum bs 1e6
ifnotinsphinx
sysstdoutwrite
rdownloaded ss currentszmb totalszmb
archivepath ospathjoindatapath ARCHIVEFILENAME
urlretrieveDOWNLOADURL filenamearchivepath
repthookprogress
ifnotinsphinx
sysstdoutwrite r
printuntarring Reuters dataset
tarfileopenarchivepath rgzextractalldatapath
printdone
parser ReutersParser
forfilename inglobospathjoindatapath sgm
fordocinparserparseopenfilename rb
yielddoc
Main
Create the vectorizer and limit the number of features to a reasonable maximum
vectorizer HashingVectorizerdecodeerrorignore nfeatures2 18
alternatesignFalse
766 Chapter 5 Examples
```

scikitlearn user guide Release 0213
Iterator over parsed Reuters SGML files
datastream streamreutersdocuments
We learn a binary classification between the acq class and all the others
acq was chosen as it is more or less evenly distributed in the Reuters
files For other datasets one should take care of creating a test set with
a realistic portion of positive instances
allclasses nparray0 1
positiveclass acq
Here are some classifiers that support the partialfit method
partialfitclassifiers
SGD SGDClassifiermaxiter5 tol1e3
Perceptron Perceptrontol1e3
NB Multinomial MultinomialNBalpha001
PassiveAggressive PassiveAggressiveClassifiertol1e3

defgetminibatchdociter size posclasspositiveclass
Extract a minibatch of examples return a tuple Xtext y
Note size is before excluding invalid docs with no topics assigned

data title nnbodyformat doc posclass indoctopics
fordocinitertoolsislicedociter size
ifdoctopics
if notlendata
returnnpasarray dtypeint npasarray dtypeint
Xtext y zip data
returnXtext npasarrayy dtypeint
defiterminibatchesdociter minibatchsize
Generator of minibatches
Xtext y getminibatchdociter minibatchsize
whilelenXtext
yieldXtext y
Xtext y getminibatchdociter minibatchsize
test data statistics
teststats ntest 0 ntestpos 0
First we hold out a number of examples to estimate accuracy
ntestdocuments 1000
tick time
Xtesttext ytest getminibatchdatastream 1000
parsingtime time tick
tick time
Xtest vectorizertransformXtesttext
vectorizingtime time tick
teststatsntest lenytest
teststatsntestpos sumytest
printTest set is ddocuments dpositive lenytest sumytest
52 Examples based on real world datasets 767

```
scikitlearn user guide Release 0213
defprogressclsname stats
Report progress information return a string
duration timetime statst0
s 20sclassifier t clsname
s ntrain6d train docs ntrainpos6d positive stats
s ntest6d test docs ntestpos6d positive teststats
s accuracy accuracy3f stats
s in 2fs 5ddocss duration statsntrain duration
returns
clsstats
forclsname inpartialfitclassifiers
stats ntrain 0 ntrainpos 0
accuracy 00 accuracyhistory 0 0 t0 timetime
runtimehistory 0 0 totalfittime 00
clsstatsclsname stats
getminibatchdatastream ntestdocuments
Discard test set
We will feed the classifier with minibatches of 1000 documents this means
we have at most 1000 docs in memory at any time The smaller the document
batch the bigger the relative overhead of the partial fit methods
minibatchsize 1000
Create the datastream that parses Reuters SGML files and iterates on
documents as a stream
minibatchiterators iterminibatchesdatastream minibatchsize
totalvecttime 00
Main loop iterate on minibatches of examples
fori Xtraintext ytrain inenumerateminibatchiterators
tick timetime
Xtrain vectorizertransformXtraintext
totalvecttime timetime tick
forclsname cls inpartialfitclassifiersitems
tick timetime
update estimator with examples in the current minibatch
clspartialfitXtrain ytrain classesallclasses
accumulate test accuracy stats
clsstatsclsnametotalfittime timetime tick
clsstatsclsnametrain Xtrainshape0
clsstatsclsnametrainpos sumytrain
tick timetime
clsstatsclsnameaccuracy clscoreXtest ytest
clsstatsclsnamepredictiontime timetime tick
acchistory clsstatsclsnameaccuracy
clsstatsclsnametrain
clsstatsclsnameaccuracyhistoryappendacchistory
runhistory clsstatsclsnameaccuracy
totalvecttime clsstatsclsnametotalfittime
768 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
clsstatsclsname runtimehistory append run history
ifi 3 0
print progress clsname clsstats clsname
ifi 3 0
print n
Out
downloading dataset once and for all into home circle scikitlearn data reuters
untarring Reuters dataset
done
Test set is 966 documents 88 positive
SGD classifier 994 train docs 121 positive 966
↳ test docs 88 positive accuracy 0907 in 074s 1342 docss
Perceptron classifier 994 train docs 121 positive 966
↳ test docs 88 positive accuracy 0922 in 074s 1338 docss
NB Multinomial classifier 994 train docs 121 positive 966
↳ test docs 88 positive accuracy 0909 in 075s 1327 docss
PassiveAggressive classifier 994 train docs 121 positive 966
↳ test docs 88 positive accuracy 0944 in 075s 1323 docss
SGD classifier 3924 train docs 496 positive 966
↳ test docs 88 positive accuracy 0957 in 220s 1784 docss
Perceptron classifier 3924 train docs 496 positive 966
↳ test docs 88 positive accuracy 0952 in 220s 1782 docss
NB Multinomial classifier 3924 train docs 496 positive 966
↳ test docs 88 positive accuracy 0917 in 221s 1778 docss
PassiveAggressive classifier 3924 train docs 496 positive 966
↳ test docs 88 positive accuracy 0967 in 221s 1776 docss
SGD classifier 6836 train docs 793 positive 966
↳ test docs 88 positive accuracy 0967 in 382s 1791 docss
Perceptron classifier 6836 train docs 793 positive 966
↳ test docs 88 positive accuracy 0950 in 382s 1790 docss
NB Multinomial classifier 6836 train docs 793 positive 966
↳ test docs 88 positive accuracy 0921 in 382s 1787 docss
PassiveAggressive classifier 6836 train docs 793 positive 966
↳ test docs 88 positive accuracy 0967 in 383s 1786 docss
SGD classifier 9597 train docs 1139 positive 966
↳ test docs 88 positive accuracy 0963 in 539s 1779 docss
Perceptron classifier 9597 train docs 1139 positive 966
↳ test docs 88 positive accuracy 0955 in 540s 1778 docss
NB Multinomial classifier 9597 train docs 1139 positive 966
↳ test docs 88 positive accuracy 0930 in 540s 1776 docss
PassiveAggressive classifier 9597 train docs 1139 positive 966
↳ test docs 88 positive accuracy 0965 in 541s 1775 docss
SGD classifier 12513 train docs 1558 positive 966
↳ test docs 88 positive accuracy 0969 in 702s 1783 docss
Perceptron classifier 12513 train docs 1558 positive 966
↳ test docs 88 positive accuracy 0843 in 702s 1783 docss
NB Multinomial classifier 12513 train docs 1558 positive 966
↳ test docs 88 positive accuracy 0936 in 702s 1781 docss
52 Examples based on real world datasets 769
```

```
scikitlearn user guide Release 0213
PassiveAggressive classifier 12513 train docs 1558 positive 966
↳test docs 88 positive accuracy 0965 in 703s 1781 docss
SGD classifier 14935 train docs 1863 positive 966
↳test docs 88 positive accuracy 0966 in 852s 1752 docss
Perceptron classifier 14935 train docs 1863 positive 966
↳test docs 88 positive accuracy 0958 in 853s 1751 docss
NB Multinomial classifier 14935 train docs 1863 positive 966
↳test docs 88 positive accuracy 0938 in 853s 1750 docss
PassiveAggressive classifier 14935 train docs 1863 positive 966
↳test docs 88 positive accuracy 0955 in 854s 1749 docss
SGD classifier 17417 train docs 2171 positive 966
↳test docs 88 positive accuracy 0960 in 998s 1745 docss
Perceptron classifier 17417 train docs 2171 positive 966
↳test docs 88 positive accuracy 0964 in 998s 1744 docss
NB Multinomial classifier 17417 train docs 2171 positive 966
↳test docs 88 positive accuracy 0943 in 999s 1743 docss
PassiveAggressive classifier 17417 train docs 2171 positive 966
↳test docs 88 positive accuracy 0960 in 999s 1743 docss
Plot results
defplotaccuracyx y xlegend
Plot accuracy as a function of x
x nparrayx
y nparrayy
plttitleClassification accuracy as a function of s xlegend
pltxlabel s xlegend
pltylabelAccuracy
pltgridTrue
pltplotx y
rcParamslegendfontsize 10
clsnames listsortedclsstatskeys
Plot accuracy evolution
pltfigure
for stats insortedclsstatsitems
Plot accuracy evolution with examples
accuracy nexamples zip statsaccuracyhistory
plotaccuracynexamples accuracy training examples
ax pltgca
axsetylim08 1
pltlegendclsnames locbest
pltfigure
for stats insortedclsstatsitems
Plot accuracy evolution with runtime
accuracy runtime zip statsruntimehistory
plotaccuracyruntime accuracy runtime s
ax pltgca
axsetylim08 1
770 Chapter 5 Examples
```


scikitlearn user guide Release 0213
pltlegendclsnames locbest
Plot fitting times
pltfigure
fig pltgcf
clsruntime statstotalfittime
forclsname stats insortedclsstatsitems
clsruntimeappendtotalvecttime
clsnamesappendVectorization
barcolors b g r c m y
ax pltsubplot111
rectangles pltbarrangelenclsnames clsruntime width05
colorbarcolors
axsetxticksnplinspace0 lenclsnames 1 lenclsnames
axsetxticklabelsclsnames fontsize10
ymax maxclsruntime 12
axsetylim0 ymax
axsetylabelruntime s
axsettitleTraining Times
defaultlabelrectangles
attach some text vi autolabel on rectangles
forrectinrectangles
height rectgetheight
axtextrectgetx rectgetwidth 2
105height 4f height
hacenter vabottom
pltsetppltxticks1 rotation30
autolabelrectangles
plttightlayout
pltshow
Plot prediction times
pltfigure
clsruntime
clsnames listsortedclsstatskeys
forclsname stats insortedclsstatsitems
clsruntimeappendstatspredictiontime
clsruntimeappendparsingtime
clsnamesappendReadParse nFeatExtr
clsruntimeappendvectorizingtime
clsnamesappendHashing nVect
ax pltsubplot111
rectangles pltbarrangelenclsnames clsruntime width05
colorbarcolors
axsetxticksnplinspace0 lenclsnames 1 lenclsnames
axsetxticklabelsclsnames fontsize8
pltsetppltxticks1 rotation30
ymax maxclsruntime 12
axsetylim0 ymax
52 Examples based on real world datasets 771

scikitlearn user guide Release 0213
axsetylabelruntime s
axsettitlePrediction Times dinstances ntestdocuments
autolabelrectangles
plttightlayout
pltshow
•
772 Chapter 5 Examples

scikitlearn user guide Release 0213

•

52 Examples based on real world datasets 773

scikitlearn user guide Release 0213

•

Total running time of the script 0 minutes 12190 seconds

53 Biclustering

Examples concerning the sklearnclusterbicluster module

Note Click here to download the full example code

531 A demo of the Spectral CoClustering algorithm

This example demonstrates how to generate a dataset and bicluster it using the Spectral CoClustering algorithm

The dataset is generated using the makebiclusters function which creates a matrix of small values and im
plants bicluster with large values The rows and columns are then shuffled and passed to the Spectral CoClustering
algorithm Rearranging the shuffled matrix to make biclusters contiguous shows how accurately the algorithm found
the biclusters

53 Biclustering 775

scikitlearn user guide Release 0213

- 53 Biclustering 777

scikitlearn user guide Release 0213

•

Out
consensus score 1000
printdoc
 Author Kemal Eren kemalkemalerencom
 License BSD (Binary Search Decision) 3 clause
import numpy as np
from matplotlib import pyplot as plt
from sklearn.datasets import make_biclusters
from sklearn.datasets import sample_generator as sg
from sklearn.cluster.bicluster import SpectralCoclustering
from sklearn.metrics import consensus_score
data, rows, columns = make_biclusters(
 shape=(300, 300), n_clusters=(5, 5), noise=5,
 shuffle=False, random_state=0)
778 Chapter 5 Examples

scikitlearn user guide Release 0213
pltmatshowdata cmappltcmBlues
plttitleOriginal dataset
data rowidx colidx sgshuffledata randomstate0
pltmatshowdata cmappltcmBlues
plttitleShuffled dataset
model SpectralCoclusteringnclusters5 randomstate0
modelfitdata
score consensusscoremodelbiclusters
rows rowidx columns colidx
printconsensus score 3formatscore
fitdata datanpargsortmodelrowlabels
fitdata fitdata npargsortmodelcolumnlabels
pltmatshowfitdata cmappltcmBlues
plttitleAfter biclustering rearranged to show biclusters
pltshow
Total running time of the script 0 minutes 0061 seconds
Note Click here to download the full example code
532 A demo of the Spectral Biclustering algorithm
This example demonstrates how to generate a checkerboard dataset and bicluster it using the Spectral Biclustering algorithm
The data is generated with the makecheckerboard function then shuffled and passed to the Spectral Biclustering algorithm The rows and columns of the shuffled matrix are rearranged to show the biclusters found by the algorithm
The outer product of the row and column label vectors shows a representation of the checkerboard structure
53 Biclustering 779

scikitlearn user guide Release 0213

-

53 Biclustering 781

scikitlearn user guide Release 0213

•

```
Out
consensus score 10
printdoc
  Author Kemal Eren kemalkemalerencom
  License BSD (Binary Search Decision) 3 clause
import numpy as np
from matplotlib import pyplotasplt
from sklearndatasets import makecheckerboard
from sklearndatasets import samplesgenerator assg
from sklearnclusterbicluster import SpectralBiclustering
from sklearnmetrics import consensussscore
nclusters 4 3
data rows columns makecheckerboard
shape300 300 nclustersnclusters noise10
shuffleFalse randomstate0
53 Biclustering 783
```

```
scikitlearn user guide Release 0213
pltmatshowdata cmappltcmBlues
plttitleOriginal dataset
data rowidx colidx sgshuffledata randomstate0
pltmatshowdata cmappltcmBlues
plttitleShuffled dataset
model SpectralBiclusteringnclustersnclusters methodlog
randomstate0
modelfitdata
score consensussscoremodelbiclusters
rows rowidx columns colidx
printconsensus score lfformatscore
fitdata datanpargsortmodelrowlabels
fitdata fitdata npargsortmodelcolumnlabels
pltmatshowfitdata cmappltcmBlues
plttitleAfter biclustering rearranged to show biclusters
pltmatshownpouterpnsortmodelrowlabels 1
npargsortmodelcolumnlabels 1
cmappltcmBlues
plttitleCheckerboard structure of rearranged data
pltshow
Total running time of the script 0 minutes 0447 seconds
Note Click here to download the full example code
533 Biclustering documents with the Spectral Coclustering algorithm
This example demonstrates the Spectral Coclustering algorithm on the twenty newsgroups dataset The 'composms
windowmisc' category is excluded because it contains many posts containing nothing but data
The TFIDF vectorized posts form a word frequency matrix which is then biclustered using Dhillon's Spectral Co
Clustering algorithm The resulting documentword biclusters indicate subsets words used more often in those subsets
documents
For a few of the best biclusters its most common document categories and its ten most important words get printed
The best biclusters are determined by their normalized cut The best words are determined by comparing their sums
inside and outside the bicluster
For comparison the documents are also clustered using MiniBatchKMeans The document clusters derived from the
biclusters achieve a better Vmeasure than clusters found by MiniBatchKMeans
Out
Vectorizing
Coclustering
Done in 410s Vmeasure 04435
MiniBatchKMeans
Done in 636s Vmeasure 03344
784 Chapter 5 Examples
```

scikitlearn user guide Release 0213
Best biclusters

bicluster 0 1957 documents 4363 words
categories 23 talkpoliticsguns 18 talkpoliticsmisc 17 scimed
words gun guns geb banks gordon clinton pitt cdt surrender veal
bicluster 1 1263 documents 3551 words
categories 27 socreligionchristian 25 talkpoliticsmideast 24 altatheism
words god jesus christians sin objective kent belief christ faith
↪moral
bicluster 2 2212 documents 2774 words
categories 18 compsysmachardware 17 compsysibmpchardware 15 comp
↪graphics
words voltage board dsp stereo receiver packages shipping circuit
↪package compression
bicluster 3 1774 documents 2629 words
categories 27 recmotorcycles 23 recautos 13 miscforsale
words bike car dod engine motorcycle ride honda bikes helmet bmw
bicluster 4 200 documents 1167 words
categories 81 talkpoliticsmideast 10 altatheism 8 socreligionchristian
words turkish armenia armenian armenians turks petch sera zuma argic
↪gvg47
from collections import defaultdict
import operator
from time import time
import numpy as np
from sklearnclusterbicluster import SpectralCoclustering
from sklearncluster import MiniBatchKMeans
from sklearndatasetstwentynewsgroups import fetch20newsgroups
from sklearnfeatureextractiontext import TfidfVectorizer
from sklearnmetricscluster import vmeasurescore
printdoc
defnumbnormalizertokens
Map all numeric tokens to a placeholder
For many applications tokens that begin with a number are not directly
useful but the fact that such a token exists can be relevant By applying
this form of dimensionality reduction some methods may perform better

returnNUMBER iftoken0isdigit elsetokenfortokenintokens
class NumberNormalizingVectorizer TfidfVectorizer
53 Biclustering 785

```
scikitlearn user guide Release 0213
defbuildtokenizerself
tokenize superbuildtokenizer
return lambda doc listnumbernormalizertokenizedoc
    exclude composmswindowsmisc
categories altatheism compgraphics
compsysibmpchardware compsysmachardware
compwindowsex miscforsale recautos
recmotorcycles recsportbaseball
recsportshockey scicrypt scielectronics
scimed scispace socreligionchristian
talkpoliticsguns talkpoliticsmideast
talkpoliticsmisc talkreligionmisc
newsgroups fetch20newsgroupscategoriescategories
ytrue newsgroupstarget
vectorizer NumberNormalizingVectorizerstopwordseenglish mindf5
cocluster SpectralCoclusteringnclusterslencategories
svdmethodarpack randomstate0
kmeans MiniBatchKMeansnclusterslencategories batchsize20000
randomstate0
printVectorizing
X vectorizerfittransformnewsgroupsdata
printCoclustering
starttime time
coclusterfitX
ycocluster coclusterrowlabels
printDone in 2fs Vmeasure 4fformat
time starttime
vmeasurescoreycocluster ytrue
printMiniBatchKMeans
starttime time
ykmeans kmeansfitpredictX
printDone in 2fs Vmeasure 4fformat
time starttime
vmeasurescoreykmeans ytrue
featurenames vectorizergetfeaturenames
documentnames listnewsgroupstargetnamesi foriinnewsgroupstarget
defbiclusterncuti
rows cols coclustergetindicesi
if notnpanyrows andnpanycols
import sys
returnsysfloatinfomax
rowcomplement npnonzeronplogicalnotcoclusterrowsi0
colcomplement npnonzeronplogicalnotcoclustercolumnsi0
    Note the following is identical to Xrows npnewaxis
    colssum but much faster in scipy 016
weight Xrows colssum
cut Xrowcomplement colssum
Xrows colcomplementsum
returncut weight
786 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
defmostcommon
Items of a defaultdict with the highest values
Like Countermostcommon in Python 27

returnsorteddictitems keyoperatoritemgetter1 reverseTrue
biclusterncuts listbiclusterncuti
foriinrangelennnewsgroutstargetnames
bestidx npargsortbiclusterncuts5
print
printBest biclusters
print
foridx cluster inenumeratebestidx
nrows ncols coclustergetshapecluster
clusterdocs clusterwords coclustergetindicescluster
if notlenclusterdocs or notlenclusterwords
continue
categories
counter defaultdictint
foriinclusterdocs
counterdocumentnamesi 1
catstring joinOf formatfloatc nrows 100 name
forname c inmostcommoncounter3
words
outofclusterdocs coclusterrowlabels cluster
outofclusterdocs npwhereoutofclusterdocs0
wordcol X clusterwords
wordscores nparraywordcolclusterdocs sumaxis0
wordcoloutofclusterdocs sumaxis0
wordscores wordscoresravel
importantwords listfeaturenamesclusterwordsi
foriinwordscoresargsort111
printbicluster documents wordsformat
idx nrows ncols
printcategories formatcatstring
printwords nformat joinimportantwords
Total running time of the script 0 minutes 13291 seconds
54 Calibration
Examples illustrating the calibration of predicted probabilities of classifiers
Note Click here to download the full example code
54 Calibration 787
```

541 Comparison of Calibration of Classifiers

Well calibrated classifiers are probabilistic classifiers for which the output of the predict_proba method can be directly interpreted as a confidence level. For instance, a well calibrated binary classifier should classify the samples such that among the samples to which it gave a predict_proba value close to 0.8, approx. 80% actually belong to the positive class.

LogisticRegression returns well calibrated predictions as it directly optimizes logloss. In contrast, the other methods return biased probabilities with different biases per method.

- GaussianNaiveBayes tends to push probabilities to 0 or 1; note the counts in the histograms. This is mainly because it makes the assumption that features are conditionally independent given the class, which is not the case in this dataset which contains 2 redundant features.
- RandomForestClassifier shows the opposite behavior: the histograms show peaks at approx. 0.2 and 0.9 probability, while probabilities close to 0 or 1 are very rare. An explanation for this is given by Niculescu-Mizil and Caruana: "Methods such as bagging and random forests that average predictions from a base set of models can have difficulty making predictions near 0 and 1 because variance in the underlying base models will bias predictions that should be near zero or one away from these values. Because predictions are restricted to the interval [0, 1], errors caused by variance tend to be one-sided: near zero and one. For example, if a model should predict p = 0 for a case, the only way bagging can achieve this is if all bagged trees predict zero. If we add noise to the trees, that bagging is averaging over this noise will cause some trees to predict values larger than 0; for this case, thus moving the average prediction of the bagged ensemble away from 0. We observe this effect most strongly with random forests because the base-level trees trained with random forests have relatively high variance due to feature subsetting." As a result, the calibration curve shows a characteristic sigmoid shape, indicating that the classifier could trust its "intuition" more and return probabilities closer to 0 or 1 typically.
- Support Vector Classification (SVC) (Support Vector Classification) shows an even more sigmoid curve than the RandomForestClassifier, which is typical for maximum-margin methods. Compare Niculescu-Mizil and Caruana, which focus on hard samples that are close to the decision boundary, the support vectors.

References

1. Predicting Good Probabilities with Supervised Learning. A. Niculescu-Mizil, R. Caruana. ICML 2005.

788 Chapter 5 Examples

```
scikitlearn user guide Release 0213
printdoc
  Author Jan Hendrik Metzen jhminformatikunibremende
  License BSD (Binary Search Decision) Style
import numpy as np
nprandomseed0
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearnnaivebayes import GaussianNB
from sklearnlinearmodel import LogisticRegression
from sklearnensemble import RandomForestClassifier
from sklearnsvm import LinearSVC
54 Calibration 789
```

```
scikitlearn user guide Release 0213
from sklearn.calibration import calibrationcurve
X y datasetsmakeclassificationnsamples100000 nfeatures20
ninformative2 nredundant2
trainsamples 100 Samples used for training the models
Xtrain Xtrainsamples
Xtest Xtrainsamples
ytrain ytrainsamples
ytest ytrainsamples
Create classifiers
lr LogisticRegressionSolverlbfgs
gnb GaussianNB
svc LinearSVCC10
rfc RandomForestClassifiernestimators100
```

```
Plot calibration plots
pltfigurefigsize10 10
ax1 pltsubplot2grid3 1 0 0 rowspan2
ax2 pltsubplot2grid3 1 2 0
ax1plot0 1 0 1 k labelPerfectly calibrated
forclf name inlr Logistic
gnb Naive Bayes
svc Support Vector Classification
rfc Random Forest
clffitXtrain ytrain
ifhasattrclf predictproba
probpos clfpredictprobaXtest 1
else use decision function
probpos clfdecisionfunctionXtest
probpos
probpos probposmin probposmax probposmin
fractionofpositives meanpredictedvalue
calibrationcurveytest probpos nbins10
ax1plotmeanpredictedvalue fractionofpositives s
labels name
ax2histprobpos range0 1 bins10 labelname
histtypestep lw2
ax1setylabelFraction of positives
ax1setylim005 105
ax1legendloclower right
ax1settitleCalibration plots reliability curve
ax2setxlabelMean predicted value
ax2setylabelCount
ax2legendlocupper center ncol2
plttightlayout
pltshow
790 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Total running time of the script 0 minutes 1028 seconds

Note [Click here to download the full example code](#)

542 Probability Calibration curves

When performing classification one often wants to predict not only the class label but also the associated probability This probability gives some kind of confidence on the prediction This example demonstrates how to display how well calibrated the predicted probabilities are and how to calibrate an uncalibrated classifier

The experiment is performed on an artificial dataset for binary classification with 100000 samples 1000 of them are used for model fitting with 20 features Of the 20 features only 2 are informative and 10 are redundant The first figure shows the estimated probabilities obtained with logistic regression Gaussian naive Bayes and Gaussian naive Bayes with both isotonic calibration and sigmoid calibration The calibration performance is evaluated with Brier score reported in the legend the smaller the better One can observe here that logistic regression is well calibrated while raw Gaussian naive Bayes performs very badly This is because of the redundant features which violate the assumption of feature independence and result in an overly confident classifier which is indicated by the typical transposed sigmoid curve

Calibration of the probabilities of Gaussian naive Bayes with isotonic regression can fix this issue as can be seen from the nearly diagonal calibration curve Sigmoid calibration also improves the brier score slightly albeit not as strongly as the nonparametric isotonic regression This can be attributed to the fact that we have plenty of calibration data such that the greater flexibility of the nonparametric model can be exploited

The second figure shows the calibration curve of a linear support vector classifier LinearSVC LinearSVC shows the opposite behavior as Gaussian naive Bayes the calibration curve has a sigmoid curve which is typical for an underconfident classifier In the case of LinearSVC this is caused by the margin property of the hinge loss which lets the model focus on hard samples that are close to the decision boundary the support vectors Both kinds of calibration can fix this issue and yield nearly identical results This shows that sigmoid calibration can deal with situations where the calibration curve of the base classifier is sigmoid eg for LinearSVC but not where it is transposed sigmoid eg Gaussian naive Bayes

scikitlearn user guide Release 0213

- Out
- Logistic
- Brier 0099
- Precision 0872
- Recall 0851
- F1 0862
- Naive Bayes
- Brier 0118
- Precision 0857
- Recall 0876
- F1 0867
- Naive Bayes Isotonic
- Brier 0098
- Precision 0883
- 54 Calibration 793

scikitlearn user guide Release 0213
Recall 0836
F1 0859
Naive Bayes Sigmoid
Brier 0109
Precision 0861
Recall 0871
F1 0866
Logistic
Brier 0099
Precision 0872
Recall 0851
F1 0862
SVC (Support Vector Classification)
Brier 0163
Precision 0872
Recall 0852
F1 0862
SVC (Support Vector Classification) Isotonic
Brier 0100
Precision 0853
Recall 0878
F1 0865
SVC (Support Vector Classification) Sigmoid
Brier 0099
Precision 0874
Recall 0849
F1 0861
printdoc
Author Alexandre Gramfort alexandregramforttelecomparistechfr
Jan Hendrik Metzen jhminformatikunibremende
License BSD (Binary Search Decision) Style
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearnnaivebayes import GaussianNB
from sklearnsvm import LinearSVC
from sklearnlinearmodel import LogisticRegression
from sklearnmetrics import brierscoreloss precisionscore recallscore
f1score
from sklearncalibration import CalibratedClassifierCV calibrationcurve
from sklearnmodelselection import traintestsplit
Create dataset of classification task with many redundant and few
informative features
794 Chapter 5 Examples


```
scikitlearn user guide Release 0213
X y datasetsmakeclassificationnsamples100000 nfeatures20
ninformative2 nredundant10
randomstate42
Xtrain Xtest ytrain ytest traintestsplitX y testsize099
randomstate42
defplotcalibrationcurveest name figindex
Plot calibration curve for est wo and with calibration
    Calibrated with isotonic calibration
isotonic CalibratedClassifierCVest cv2 methodisotonic
    Calibrated with sigmoid calibration
sigmoid CalibratedClassifierCVest cv2 methodsigmoid
    Logistic regression with no calibration as baseline
lr LogisticRegressionC1 solverlbfgs
fig pltfigurefigindex figsize10 10
ax1 pltsubplot2grid3 1 0 0 rowspan2
ax2 pltsubplot2grid3 1 2 0
ax1plot0 1 0 1 k labelPerfectly calibrated
forclf name inlr Logistic
est name
isotonic name Isotonic
sigmoid name Sigmoid
clffitXtrain ytrain
ypred clfpredictXtest
ifhasattrclf predictproba
probpos clfpredictprobaXtest 1
else use decision function
probpos clfdecisionfunctionXtest
probpos
probpos probposmin probposmax probposmin
clfscore brierscorelossytest probpos poslabelymax
prints name
printtBrier13f clfscore
printtPrecision 13f precisionscoreytest ypred
printtRecall 13f recallscoreytest ypred
printtF113fn f1scoreytest ypred
fractionofpositives meanpredictedvalue
calibrationcurveytest probpos nbins10
ax1plotmeanpredictedvalue fractionofpositives s
labels13f name clfscore
ax2histprobpos range0 1 bins10 labelname
histtypestep lw2
ax1setylabelFraction of positives
ax1setylim005 105
ax1legendloclower right
ax1settitleCalibration plots reliability curve
54 Calibration 795
```

scikitlearn user guide Release 0213

ax2setxlabelMean predicted value

ax2setylabelCount

ax2legendlocupper center ncol2

plt.tight_layout

Plot calibration curve for Gaussian Naive Bayes

plotcalibrationcurveGaussianNB Naive Bayes 1

Plot calibration curve for Linear SVC (Support Vector Classification)

plotcalibrationcurveLinearSVCmaxiter10000 SVC (Support Vector Classification) 2

plt.show

Total running time of the script 0 minutes 1993 seconds

Note Click here to download the full example code

543 Probability calibration of classifiers

When performing classification you often want to predict not only the class label but also the associated probability

This probability gives you some kind of confidence on the prediction However not all classifiers provide well

calibrated probabilities some being overconfident while others being underconfident Thus a separate calibration

of predicted probabilities is often desirable as a postprocessing This example illustrates two different methods for

this calibration and evaluates the quality of the returned probabilities using Brier's score see https://en.wikipedia.org/wiki/Brier_score

wikiBrierscore

Compared are the estimated probability using a Gaussian naive Bayes classifier without calibration with a sigmoid

calibration and with a nonparametric isotonic calibration One can observe that only the nonparametric model is

able to provide a probability calibration that returns probabilities close to the expected 0.5 for most of the samples

belonging to the middle cluster with heterogeneous labels This results in a significantly improved Brier score

796 Chapter 5 Examples

scikitlearn user guide Release 0213

•

Out

Brier scores the smaller the better

No calibration 0104

With isotonic calibration 0084

With sigmoid calibration 0109

printdoc

Author Mathieu Blondel mathieumblondelorg

Alexandre Gramfort alexandregamforttelecomparistechfr

Balazs Kegl balazskeglgmailcom

Jan Hendrik Metzen jhminformatikunibremende

License BSD (Binary Search Decision) Style

import numpy as np

import matplotlib.pyplot as plt

from matplotlib import cm

from sklearn.datasets import makeblobs

from sklearn.naivebayes import GaussianNB

from sklearn.metrics import brierscoreloss

798 Chapter 5 Examples

```
scikitlearn user guide Release 0213
from sklearncalibration import CalibratedClassifierCV
from sklearnmodelselection import traintestsplit
nsamples 50000
nbins 3 use 3 bins for calibrationcurve as we have 3 clusters here
Generate 3 blobs with 2 classes where the second blob contains
half positive samples and half negative samples Probability in this
blob is therefore 0.5
centers 5 5 0 0 5 5
X y makeblobsnsamplesnsamples nfeatures2 clusterstd10
centerscenters shuffleFalse randomstate42
y nsamples 2 0
y nsamples 2 1
sampleweight nprandomRandomState42randyshape0
split train test for calibration
Xtrain Xtest ytrain ytest swtrain swtest
traintestsplitX y sampleweight testsize0.9 randomstate42
Gaussian NaiveBayes with no calibration
clf GaussianNB
clf.fit(Xtrain ytrain GaussianNB itself does not support sampleweights
probposclf clf.predict_proba(Xtest 1
Gaussian NaiveBayes with isotonic calibration
clfisotonic CalibratedClassifierCVclf cv2 methodisotonic
clfisotonic.fit(Xtrain ytrain swtrain
probposisotonic clfisotonic.predict_proba(Xtest 1
Gaussian NaiveBayes with sigmoid calibration
clfsigmoid CalibratedClassifierCVclf cv2 methodsigmoid
clfsigmoid.fit(Xtrain ytrain swtrain
probpossigmoid clfsigmoid.predict_proba(Xtest 1
printBrier scores the smaller the better
clfscore brierscorelossytest probposclf swtest
printNo calibration 13f clfscore
clfisotonicscore brierscorelossytest probposisotonic swtest
printWith isotonic calibration 13f clfisotonic.score
clfsigmoidscore brierscorelossytest probpossigmoid swtest
printWith sigmoid calibration 13f clfsigmoid.score

Plot the data and the predicted probabilities
plt.figure
yunique np.unique
colors cm.rainbow(nplinspace(0,10 yunique.size
for thisy color in zip(yunique colors
thisX Xtrainytrain thisy
thissw swtrainytrain thisy
plt.scatter(thisX 0 thisX 1 thissw 50
ccolor np.newaxis
54 Calibration 799
```

scikitlearn user guide Release 0213

alpha05 edgecolork

labelClass s thisy

pltlegendlocbest

plttitleData

pltfigure

order nplexsortprobposclf

pltplotprobposclforder r labelNo calibration 13f clfscore

pltplotprobposisotonicorder g linewidth3

labelIsotonic calibration 13f clfisotonicscore

pltplotprobpossigmoidorder b linewidth3

labelSigmoid calibration 13f clfsigmoidscore

pltplotnplinspace0 ytestsize 5112

ytestorderreshape25 1mean1

k linewidth3 labelrEmpirical

pltylim005 105

pltxlabelInstances sorted according to predicted probability

uncalibrated GNB

pltylabelPy1

pltlegendlocupper left

plttitleGaussian naive Bayes probabilities

pltshow

Total running time of the script 0 minutes 0108 seconds

Note Click here to download the full example code

544 Probability Calibration for 3class classification

This example illustrates how sigmoid calibration changes predicted probabilities for a 3class classification problem

Illustrated is the standard 2simplex where the three corners correspond to the three classes Arrows point from the

probability vectors predicted by an uncalibrated classifier to the probability vectors predicted by the same classifier

after sigmoid calibration on a holdout validation set Colors indicate the true class of an instance red class 1 green

class 2 blue class 3

The base classifier is a random forest classifier with 25 base estimators trees If this classifier is trained on all 800

training datapoints it is overly confident in its predictions and thus incurs a large logloss Calibrating an identical

classifier which was trained on 600 datapoints with method'sigmoid' on the remaining 200 datapoints reduces the

confidence of the predictions ie moves the probability vectors from the edges of the simplex towards the center

This calibration results in a lower logloss Note that an alternative would have been to increase the number of base

estimators which would have resulted in a similar decrease in logloss

800 Chapter 5 Examples


```
scikitlearn user guide Release 0213
•
Out
Logloss of
uncalibrated classifier trained on 800 datapoints 1280
classifier trained on 600 datapoints and calibrated on 200 datapoint 0534
printdoc
  Author Jan Hendrik Metzen jhminformatikunibremende
  License BSD (Binary Search Decision) Style
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import makeblobs
from sklearn.ensemble import RandomForestClassifier
from sklearn.calibration import CalibratedClassifierCV
from sklearn.metrics import logloss
nprandomseed0
802 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
Generate data
X y makeblobsnsamples1000 nfeatures2 randomstate42
clusterstd50
Xtrain ytrain X600 y600
Xvalid yvalid X600800 y600800
Xtraininvalid ytraininvalid X800 y800
Xtest ytest X800 y800
Train uncalibrated random forest classifier on whole train and validation
data and evaluate on test data
clf RandomForestClassifiernestimators25
clffitXtraininvalid ytraininvalid
clfprobs clfpredictprobaXtest
score loglossytest clfprobs
Train random forest classifier calibrate on validation data and evaluate
on test data
clf RandomForestClassifiernestimators25
clffitXtrain ytrain
clfprobs clfpredictprobaXtest
sigclf CalibratedClassifierCVclf methodsigmoid cvprefit
sigclffitXvalid yvalid
sigclfprobs sigclfpredictprobaXtest
sigscore loglossytest sigclfprobs
Plot changes in predicted probabilities via arrows
pltfigure
colors r g b
foriinrangeclfprobsshape0
pltarrowclfprobsi 0 clfprobsi 1
sigclfprobsi 0 clfprobsi 0
sigclfprobsi 1 clfprobsi 1
colorcolorsytesti headwidth1e2
Plot perfect predictions
pltplot10 00 ro ms20 labelClass 1
pltplot00 10 go ms20 labelClass 2
pltplot00 00 bo ms20 labelClass 3
Plot boundaries of unit simplex
pltplot00 10 00 00 00 00 10 00 k labelSimplex
Annotate points on the simplex
pltannotaterfrac13 frac13 frac13
xy103 103 xytext103 23 xycoordsdata
arrowpropsdictfacecolorblack shrink005
horizontalalignmentcenter verticalalignmentcenter
pltplot103 103 ko ms5
pltannotaterfrac12 0 frac12
xy5 0 xytext5 1 xycoordsdata
arrowpropsdictfacecolorblack shrink005
horizontalalignmentcenter verticalalignmentcenter
pltannotater0 frac12 frac12
xy0 5 xytext1 5 xycoordsdata
arrowpropsdictfacecolorblack shrink005
horizontalalignmentcenter verticalalignmentcenter
pltannotaterfrac12 frac12 0
54 Calibration 803
```

scikitlearn user guide Release 0213

xy5 5 xytext6 6 xycoordsdata
arrowpropsdictfacecolorblack shrink005
horizontalalignmentcenter verticalalignmentcenter
pltannotater0 0 1

xy0 0 xytext1 1 xycoordsdata
arrowpropsdictfacecolorblack shrink005
horizontalalignmentcenter verticalalignmentcenter
pltannotater1 0 0

xy1 0 xytext1 1 xycoordsdata
arrowpropsdictfacecolorblack shrink005
horizontalalignmentcenter verticalalignmentcenter
pltannotater0 1 0

xy0 1 xytext1 1 xycoordsdata
arrowpropsdictfacecolorblack shrink005
horizontalalignmentcenter verticalalignmentcenter

Add grid
pltgridFalse

forxin00 01 02 03 04 05 06 07 08 09 10

pltplot0 x x 0 k alpha02
pltplot0 0 1x2 x x 1x2 k alpha02
pltplotx x 1x2 0 0 1x2 k alpha02

plttitleChange of predicted probabilities after sigmoid calibration
pltxlabelProbability class 1
pltylabelProbability class 2
pltxlim005 105
pltylim005 105
pltlegendlocbest

printLogloss of
printuncalibrated classifier trained on 800 datapoints 3f
score

printclassifier trained on 600 datapoints and calibrated on
200 datapoint 3f sigscore

Illustrate calibrator
pltfigure

generate grid over 2simplex
p1d nplinspace0 1 20
p0 p1 npmeshgridp1d p1d
p2 1 p0 p1
p npcp0ravel p1ravel p2ravel
p pp 2 0

calibratedclassifier sigclfcalibratedclassifiers0
prediction npvstackcalibratorpredictthisp
forcalibrator thisp in

zipcalibratedclassifiercalibrators pTT
prediction predictionsumaxis1 None

Plot modifications of calibrator
foriinrangepredictionsshape0

pltarrowpi 0 pi 1
predictioni 0 pi 0 predictioni 1 pi 1
headwidth1e2 colorcolorsnpargmaxpi

Plot boundaries of unit simplex
pltplot00 10 00 00 00 00 10 00 k labelSimplex

804 Chapter 5 Examples

```
scikitlearn user guide Release 0213
pltgridFalse
forxin00 01 02 03 04 05 06 07 08 09 10
pltplot0 x x 0 k alpha02
pltplot0 0 1x2 x x 1x2 k alpha02
pltplotx x 1x2 0 0 1x2 k alpha02
plttitleIllustration of sigmoid calibrator
pltxlabelProbability class 1
pltylabelProbability class 2
pltxlim005 105
pltylim005 105
pltshow
Total running time of the script 0 minutes 0318 seconds
55 Classification
General examples about classification algorithms
Note Click here to download the full example code
551 Recognizing handwritten digits
An example showing how the scikitlearn can be used to recognize images of handwritten digits
This example is commented in the tutorial section of the user manual
55 Classification 805
```

scikitlearn user guide Release 0213

Out

Classification report for classifier SVCgamma0001

precision recall f1score support

| | | | | |
|---|-----|-----|-----|----|
| 0 | 100 | 099 | 099 | 88 |
| 1 | 099 | 097 | 098 | 91 |
| 2 | 099 | 099 | 099 | 86 |
| 3 | 098 | 087 | 092 | 91 |
| 4 | 099 | 096 | 097 | 92 |
| 5 | 095 | 097 | 096 | 91 |
| 6 | 099 | 099 | 099 | 91 |
| 7 | 096 | 099 | 097 | 89 |
| 8 | 094 | 100 | 097 | 88 |
| 9 | 093 | 098 | 095 | 92 |

accuracy 097 899

macro avg 097 097 097 899

weighted avg 097 097 097 899

Confusion matrix

| | | | | | | | | |
|----|----|----|----|---|---|---|---|---|
| 87 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 88 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 85 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 79 | 0 | 3 | 0 | 4 | 5 |

806 Chapter 5 Examples

scikitlearn user guide Release 0213

0 0 0 0 88 0 0 0 4
0 0 0 0 0 88 1 0 0 2
0 1 0 0 0 0 90 0 0 0
0 0 0 0 0 1 0 88 0 0
0 0 0 0 0 0 0 0 88 0
0 0 0 1 0 1 0 0 0 90

printdoc

Author Gael Varoquaux gael dot varoquaux at normalesup dot org

License BSD (Binary Search Decision) 3 clause

Standard scientific Python imports

import matplotlib.pyplot as plt

Import datasets classifiers and performance metrics

from sklearn import datasets svm metrics

The digits dataset

digits datasetsloaddigits

The data that we are interested in is made of 8x8 images of digits lets
have a look at the first 4 images stored in the images attribute of the
dataset If we were working from image files we could load them using
matplotlib.pyplot.imread Note that each image must have the same size For these
images we know which digit they represent it is given in the target of
the dataset

imagesandlabels listzipdigitsimages digitstarget

forindex image label inenumerateimagesandlabels4

plt.subplot2 4 index 1

plt.axisoff

plt.imshowimage cmapplt.cm.grayr interpolationnearest

plt.titleTraining i label

To apply a classifier on this data we need to flatten the image to

turn the data in a samples feature matrix

nsamples len(digitsimages)

data digitsimages.reshape(samples 1)

Create a classifier a support vector classifier

classifier svm.SVC(gamma=0.001)

We learn the digits on the first half of the digits

classifier.fit(data[:samples // 2], digitstarget[:samples // 2])

Now predict the value of the digit on the second half

expected digitstarget[samples // 2:]

predicted classifier.predict(data[samples // 2:])

print(Classification report for classifier: %s\n\n" % classifier)

classifier.metrics(classification_report(expected, predicted))

print(Confusion matrix: %s\n\n" % metrics.confusion_matrix(expected, predicted))

55 Classification 807

```
scikitlearn user guide Release 0213
imagesandpredictions listzipdigitsimagesnsamples 2 predicted
forindex image prediction inenumerateimagesandpredictions4
pltsubplot2 4 index 5
pltaxisoff
pltimshowimage cmappltcmgrayr interpolationnearest
plttitlePrediction i prediction
pltshow
Total running time of the script 0 minutes 0237 seconds
Note Click here to download the full example code
552 Normal and Shrinkage Linear Discriminant Analysis for classification
Shows how shrinkage improves classification
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import makeblobs
from sklearn.discriminantanalysis import LinearDiscriminantAnalysis
808 Chapter 5 Examples
```

scikitlearn user guide Release 0213

ntrain 20 samples for training

ntest 200 samples for testing

naverages 50 how often to repeat classification

nfeaturesmax 75 maximum number of features

step 4 step size for the calculation

defgeneratedatansamples nfeatures

Generate random blobish data with noisy features

This returns an array of input data with shape nsamples nfeatures

and an array of nsamples target labels

Only one feature contains discriminative information the other features

contain only noise

X y makeblobsnsamplesnsamples nfeatures1 centers2 2

add nondiscriminative features

ifnfeatures 1

X npstackX nprandomrandnnsamples nfeatures 1

returnX y

accclf1 accclf2

nfeaturesrange range1 nfeaturesmax 1 step

forfeatures innfeaturesrange

scoreclf1 scoreclf2 0 0

forinrangenaverages

X y generatedatantrain nfeatures

clf1 LinearDiscriminantAnalysis solverlsqr shrinkageautofitX y

clf2 LinearDiscriminantAnalysis solverlsqr shrinkageNonefitX y

X y generatedatantest nfeatures

scoreclf1 clf1scoreX y

scoreclf2 clf2scoreX y

accclf1appendscoreclf1 naverages

accclf2appendscoreclf2 naverages

featuressamplesratio nparraynfeaturesrange ntrain

pltplotfeaturessamplesratio accclf1 linewidth2

labelLinear Discriminant Analysis with shrinkage colornavy

pltplotfeaturessamplesratio accclf2 linewidth2

labelLinear Discriminant Analysis colorgold

pltxlabelnfeatures nsamples

pltylabelClassification accuracy

pltlegendloc1 propsize 12

pltsuptitleLinear Discriminant Analysis vs

shrinkage Linear Discriminant Analysis 1 discriminative feature

pltshow

Total running time of the script 0 minutes 6160 seconds

55 Classification 809

scikitlearn user guide Release 0213

Note [Click here to download the full example code](#)

553 Plot classification probability

Plot the classification probability for different classifiers We use a 3 class dataset and we classify it with a Support Vector classifier L1 and L2 penalized logistic regression with either a OneVsRest or multinomial setting and Gaussian process classification

Linear SVC (Support Vector Classification) is not a probabilistic classifier by default but it has a builtin calibration option enabled in this example `probability=True`

The logistic regression with OneVsRest is not a multiclass classifier out of the box As a result it has more trouble in separating class 2 and 3 than the other estimators

810 Chapter 5 Examples


```
scikitlearn user guide Release 0213
Out
Accuracy train for L1 logistic 833
Accuracy train for L2 logistic Multinomial 827
Accuracy train for L2 logistic OvR 793
Accuracy train for Linear SVC (Support Vector Classification) 820
Accuracy train for GPC 827
printdoc
Author Alexandre Gramfort alexandregramfortinriafr
License BSD (Binary Search Decision) 3 clause
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import accuracyscore
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC (Support Vector Classification)
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF (Radial Basis Function)
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data[:, 0:2] # we only take the first two features for visualization
y = iris.target
n_features = X.shape[1]
C = 10
kernel = 10 * RBF(10) # for GPC
# Create different classifiers
classifiers = {}
# L1 logistic
LogisticRegression(C=C, penalty='l1', solver='saga',
                    multi_class='multinomial', max_iter=10000)
# L2 logistic Multinomial
LogisticRegression(C=C, penalty='l2', solver='saga',
                    multi_class='multinomial', max_iter=10000)
# L2 logistic OvR
LogisticRegression(C=C, penalty='l2', solver='saga',
                    multi_class='ovr', max_iter=10000)
# Linear SVC (Support Vector Classification)
SVC(kernel='linear', C=C, probability=True, random_state=0)
# GPC
GaussianProcessClassifier(kernel=kernel, n_classifiers=len(classifiers))
812 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
pltfigurefigsize3 2 nclassifiers 2
pltsubplotsadjustbottom2 top95
xx nplinspace3 9 100
yy nplinspace1 5 100T
xx yy npmeshgridxx yy
Xfull npcxxravel yy.ravel
forindex name classifier in enumerate(classifiers.items)
classifierfitX y
ypred classifierpredictX
accuracy accuracyscorey ypred
printAccuracy train for s01f name accuracy 100
View probabilities
probas classifierpredictprobaXfull
nclasses npuniqueypredsize
forkinrangenclasses
pltsubplotnclassifiers nclasses index nclasses k 1
plttitleClass d k
ifk 0
pltlabelname
imshowhandle pltimshowprobas kreshape100 100
extent3 9 1 5 originlower
pltxticks
pltyticks
idx ypred k
ifidxany
pltscatterXidx 0 Xidx 1 markero cw edgecolork
ax pltaxes015 004 07 005
plttitleProbability
pltcolorbarimshowhandle caxax orientationhorizontal
pltshow
Total running time of the script 0 minutes 1409 seconds
Note Click here to download the full example code
554 Classifier comparison
A comparison of a several classifiers in scikitlearn on synthetic datasets The point of this example is to illustrate
the nature of decision boundaries of different classifiers This should be taken with a grain of salt as the intuition
conveyed by these examples does not necessarily carry over to real datasets
Particularly in highdimensional spaces data can more easily be separated linearly and the simplicity of classifiers
such as naive Bayes and linear SVMs might lead to better generalization than is achieved by other classifiers
The plots show training points in solid colors and testing points semitransparent The lower right shows the classifi
cation accuracy on the test set
55 Classification 813
```

```
scikitlearn user guide Release 0213
printdoc
Code source Gaël Varoquaux
Andreas Müller
Modified for documentation by Jaques Grobler
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import make_moons, make_circles, make_classification
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC (Support Vector Classification)
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF (Radial Basis Function)
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis

h = 0.2 # step size in the mesh

names = ['Nearest Neighbors', 'Linear SVM', 'RBF (Radial Basis Function) SVM', 'Gaussian Process',
         'Decision Tree', 'Random Forest', 'Neural Net', 'AdaBoost',
         'Naive Bayes', 'QDA',
         'classifiers',
         'KNeighborsClassifier3',
         'SVCkernel', 'linear C0025',
         'SVCgamma2 C1',
         'GaussianProcessClassifier10', 'RBF10',
         'DecisionTreeClassifiermaxdepth5',
         'RandomForestClassifiermaxdepth5', 'n_estimators10', 'max_features1',
         'MLPClassifieralpha1', 'max_iter1000',
         'AdaBoostClassifier',
         'GaussianNB',
         'QuadraticDiscriminantAnalysis',
         'X y', 'make_classification', 'n_features2', 'n_redundant0', 'n_informative2']

814 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
randomstate1 nclustersperclass1
rng np.random.RandomState2
X 2rnguniformsizeXshape
linearlyseparable X y
datasets makemoonsnoise03 randomstate0
makecirclesnoise02 factor05 randomstate1
linearlyseparable

figure plt.figure(figsize=27 9)
i 1
    iterate over datasets
    for ds in enumerate(datasets):
        preprocess dataset split into training and test part
        X y ds
        X StandardScaler.fit_transform(X)
        Xtrain Xtest ytrain ytest
        train_test_split(X y test_size=4 random_state=42)
        xmin xmax X[0]min 5 X[0]max 5
        ymin ymax X[1]min 5 X[1]max 5
        xx yy np.meshgrid(np.arange(xmin xmax h)
                           np.arange(ymin ymax h))
        just plot the dataset first
        cm plt.cm.RdBu
        cm_bright ListedColormap(FF0000 0000FF)
        ax plt.subplot(2 2 datasets len(classifiers) 1 i)
        if ds_cnt == 0:
            ax.set_title('Input data')
            Plot the training points
            ax.scatter(Xtrain[0] Xtrain[1] cytrain cmap=cm_bright)
            edgecolors=k
            Plot the testing points
            ax.scatter(Xtest[0] Xtest[1] cytest cmap=cm_bright alpha=0.6)
            edgecolors=k
            ax.set_xlim(xmin xmax)
            ax.set_ylim(ymin ymax)
            ax.set_xticks()
            ax.set_yticks()
        i 1
            iterate over classifiers
            for name, clf in zip(names, classifiers):
                ax = plt.subplot(2 2 datasets len(classifiers) 1 i)
                clf.fit(Xtrain ytrain)
                score = clf.score(Xtest ytest)
                Plot the decision boundary For that we will assign a color to each
                point in the mesh xmin xmax ymin ymax
                if hasattr(clf, 'decision_function'):
                    Z = clf.decision_function(np.cxx.ravel yy.ravel)
                else:
                    Z = clf.predict_proba(np.cxx.ravel yy.ravel)[1]
                Put the result into a color plot
                Z = Z.reshape(xxshape)
55 Classification 815
```

```
scikitlearn user guide Release 0213
axcontourfxx yy Z cmapcm alpha8
Plot the training points
axscatterXtrain 0 Xtrain 1 cytrain cmapcmbright
edgecolorsk
Plot the testing points
axscatterXtest 0 Xtest 1 cytest cmapcmbright
edgecolorsk alpha06
axsetxlimxxmin xxmax
axsetylimyymin yymax
axsetxticks
axsetyticks
ifds cnt 0
axsettitlename
axtextxxmax 3 yymin 3 2f scorelstrip0
size15 horizontalalignmentright
i 1
plttightlayout
pltshow
Total running time of the script 0 minutes 5178 seconds
Note Click here to download the full example code
555 Linear and Quadratic Discriminant Analysis with covariance ellipsoid
This example plots the covariance ellipsoids of each class and decision boundary learned by LDA and QDA The
ellipsoids display the double standard deviation for each class With LDA the standard deviation is the same for all
the classes while each class has its own standard deviation with QDA
816 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdoc
from scipy import linalg
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
from matplotlib import colors
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis

Colormap
cmap colors.LinearSegmentedColormap
redblueclasses
red 0 1 1 1 07 07
green 0 07 07 1 07 07
blue 0 07 07 1 1 1
plt.cm.register_cmap(cmapcmapcmap

Generate datasets
55 Classification 817
```

```
scikitlearn user guide Release 0213
defdatasetfixedcov
Generate 2 Gaussians samples with the same covariance matrix
n dim 300 2
nprandomseed0
C nparray0 023 083 23
X nprnpdotnprandomrandnn dim C
npdotnprandomrandnn dim C nparray1 1
y npstacknpzerosn nponesn
returnX y
defdatasetcov
Generate 2 Gaussians samples with different covariance matrices
n dim 300 2
nprandomseed0
C nparray0 1 25 7 2
X nprnpdotnprandomrandnn dim C
npdotnprandomrandnn dim CT nparray1 4
y npstacknpzerosn nponesn
returnX y
```

```
Plot functions
defplotdataalda X y ypred figindex
splot pltsubplot2 2 figindex
iffigindex 1
plttitleLinear Discriminant Analysis
pltlabelData with nfixed covariance
eliffigindex 2
plttitleQuadratic Discriminant Analysis
eliffigindex 3
pltlabelData with nvarying covariances
tp y ypred True Positive
tp0 tp1 tpy 0 tpy 1
X0 X1 Xy 0 Xy 1
X0tp X0fp X0tp0 X0tp0
X1tp X1fp X1tp1 X1tp1
class 0 dots
pltscatterX0tp 0 X0tp 1 marker colored
pltscatterX0fp 0 X0fp 1 markerx
s20 color990000 dark red
class 1 dots
pltscatterX1tp 0 X1tp 1 marker colorblue
pltscatterX1fp 0 X1fp 1 markerx
s20 color000099 dark blue
class 0 and 1 areas
nx ny 200 100
xmin xmax pltxlim
ymin ymax pltylim
xx yy npmeshgridnplinspacexmin xmax nx
nplinspaceymin ymax ny
Z ldapredictprobanpcxxravel yy.ravel
Z Z 1reshapexxshape
818 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
pltplotcolormeshxx yy Z cmapredblueclasses
normcolorsNormalize0 1 zorder0
pltcontourxx yy Z 05 linewidths2 colorswhite
means
pltplotldameans00 ldameans01
coloryellow markersize15 markeredgecolorgrey
pltplotldameans10 ldameans11
coloryellow markersize15 markeredgecolorgrey
returnsplot
defplotellipsesplot mean cov color
v w linalgeighcov
u w0 linalgnormw0
angle nparctanu1 u0
angle 180 angle nppl convert to degrees
filled Gaussian at 2 standard deviation
ell mplpatchesEllipsemmean 2 v005 2v105
180 angle facecolorcolor
edgecolorblack linewidth2
ellsetclipboxsplotbbox
ellsetalpha02
splotaddartistell
splotsetxticks
splotsetyticks
defplotldacovlda splot
plotellipsesplot ldameans0 ldacovariance red
plotellipsesplot ldameans1 ldacovariance blue
defplotqdacovqda splot
plotellipsesplot qdameans0 qdacovariance0 red
plotellipsesplot qdameans1 qdacovariance1 blue
pltfigurefigsize10 8 facecolorwhite
pltstitleLinear Discriminant Analysis vs Quadratic Discriminant Analysis
y098 fontsize15
fori X y inenumerateddatasetfixedcov datasetcov
Linear Discriminant Analysis
lda LinearDiscriminantAnalysisissolversvd storecovarianceTrue
ypred ldafitX ypredictX
splot plotdatalda X y ypred figindex2 i 1
plotldacovlda splot
pltaxistight
Quadratic Discriminant Analysis
qda QuadraticDiscriminantAnalysisstorecovarianceTrue
ypred qdafitX ypredictX
splot plotdataqda X y ypred figindex2 i 2
plotqdacovqda splot
pltaxistight
plttightlayout
pltsubplotsadjusttop092
55 Classification 819
```

scikitlearn user guide Release 0213
pltshow
Total running time of the script 0 minutes 0231 seconds
56 Clustering
Examples concerning the sklearncluster module
Note [Click here to download the full example code](#)
561 Feature agglomeration
These images how similar features are merged together using feature agglomeration
printdoc
Code source Gaël Varoquaux
Modified for documentation by Jaques Grobler
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets cluster
from sklearnfeatureextractionimage import gridtograph
digits datasetsloaddigits
images digitsimages
X npreshapeimages lenimages 1
820 Chapter 5 Examples

scikitlearn user guide Release 0213
connectivity gridtograph images0shape
agglo clusterFeatureAgglomerationconnectivityconnectivity
nclusters32
agglofitX
Xreduced agglotransformX
Xrestored aggloinversetransformXreduced
imagesrestored npreshapeXrestored imagesshape
pltfigure1 figsize4 35
pltclf
pltsubplotsadjustleft01 right99 bottom01 top91
fori in range4
pltsubplot3 4 i 1
pltimshowimagesi cmappltcmgray vmax16 interpolationnearest
pltxticks
pltyticks
ifi 1
plttitleOriginal data
pltsubplot3 4 4 i 1
pltimshowimagesrestoredi cmappltcmgray vmax16
interpolationnearest
ifi 1
plttitleAgglomerated data
pltxticks
pltyticks
pltsubplot3 4 10
pltimshownpreshapeagglolabels images0shape
interpolationnearest cmappltcmnipy spectral
pltxticks
pltyticks
plttitleLabels
pltshow
Total running time of the script 0 minutes 0174 seconds
Note Click here to download the full example code
562 A demo of the meanshift clustering algorithm
Reference
Dorin Comaniciu and Peter Meer “Mean Shift A robust approach toward feature space analysis” IEEE Transactions
on Pattern Analysis and Machine Intelligence 2002 pp 603619
56 Clustering 821

```
scikitlearn user guide Release 0213
Out
number of estimated clusters 3
printdoc
import numpy as np
from sklearncluster import MeanShift estimatebandwidth
from sklearndatasetssamplesgenerator import makeblobs

Generate sample data
centers 1 1 1 1 1 1
X makeblobsnsamples10000 centerscenters clusterstd06

Compute clustering with MeanShift
The following bandwidth can be automatically detected using
bandwidth estimatebandwidthX quantile02 nsamples500
822 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
ms MeanShiftbandwidthbandwidth binseedingTrue
msfitX
labels mslabels
clustercenters msclustercenters
labelsunique npuniquelabels
nclusters lenlabelsunique
printnumber of estimated clusters d nclusters
```

```
Plot result
import matplotlib.pyplot as plt
from itertools import cycle
pltfigure1
pltclf
colors cyclebgrcmykbgrcmykbgrcmykbgrcmyk
fork colinziprangencusters colors
mymembers labels k
clustercenter clustercentersk
pltplotXmymembers 0 Xmymembers 1 col
pltplotclustercenter0 clustercenter1 o markerfacecolorcol
markeredgecolork markersize14
plttitleEstimated number of clusters d nclusters
pltshow
```

Total running time of the script 0 minutes 0397 seconds

Note [Click here to download the full example code](#)

563 Demonstration of kmeans assumptions

This example is meant to illustrate situations where kmeans will produce unintuitive and possibly unexpected clusters
In the first three plots the input data does not conform to some implicit assumption that kmeans makes and undesirable clusters are produced as a result In the last plot kmeans returns intuitive clusters despite unevenly sized blobs

```
scikitlearn user guide Release 0213
printdoc
  Author Phil Roth mrphilrothgmailcom
  License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearncluster import KMeans
from sklearndatasets import makeblobs
pltfigurefigsize12 12
nsamples 1500
randomstate 170
824 Chapter 5 Examples
```

scikitlearn user guide Release 0213

X y makeblobsnsamplesnsamples randomstaterandomstate
Incorrect number of clusters
ypred KMeansnclusters2 randomstaterandomstatefitpredictX
pltsubplot221
pltscatterX 0 X 1 cypred
plttitleIncorrect Number of Blobs
Anisotropically distributed data
transformation 060834549 063667341 040887718 085253229
Xaniso npdotX transformation
ypred KMeansnclusters3 randomstaterandomstatefitpredictXaniso
pltsubplot222
pltscatterXaniso 0 Xaniso 1 cypred
plttitleAnisotropically Distributed Blobs
Different variance
Xvaried yvaried makeblobsnsamplesnsamples
clusterstd10 25 05
randomstaterandomstate
ypred KMeansnclusters3 randomstaterandomstatefitpredictXvaried
pltsubplot223
pltscatterXvaried 0 Xvaried 1 cypred
plttitleUnequal Variance
Unevenly sized blobs
Xfiltered npvstackXy 0500 Xy 1100 Xy 210
ypred KMeansnclusters3
randomstaterandomstatefitpredictXfiltered
pltsubplot224
pltscatterXfiltered 0 Xfiltered 1 cypred
plttitleUnevenly Sized Blobs
pltshow
Total running time of the script 0 minutes 0163 seconds
Note Click here to download the full example code
564 Online learning of a dictionary of parts of faces
This example uses a large dataset of faces to learn a set of 20 x 20 images patches that constitute faces
From the programming standpoint it is interesting because it shows how to use the online API (Application Programming Interface) of the scikitlearn
to process a very large dataset by chunks The way we proceed is that we load an image at a time and extract
randomly 50 patches from this image Once we have accumulated 500 of these patches using 10 images we run the
partialfit method of the online KMeans object MiniBatchKMeans
The verbose setting on the MiniBatchKMeans enables us to see that some clusters are reassigned during the successive
calls to partialfit This is because the number of patches that they represent has become too low and it is better to
choose a random new cluster
56 Clustering 825

scikitlearn user guide Release 0213
Out
Learning the dictionary
Partial fit of 100 out of 2400
Partial fit of 200 out of 2400
MiniBatchKMeans Reassigning 16 cluster centers
Partial fit of 300 out of 2400
Partial fit of 400 out of 2400
Partial fit of 500 out of 2400
Partial fit of 600 out of 2400
Partial fit of 700 out of 2400
Partial fit of 800 out of 2400
Partial fit of 900 out of 2400
Partial fit of 1000 out of 2400
Partial fit of 1100 out of 2400
Partial fit of 1200 out of 2400
Partial fit of 1300 out of 2400
Partial fit of 1400 out of 2400
Partial fit of 1500 out of 2400
Partial fit of 1600 out of 2400
Partial fit of 1700 out of 2400
Partial fit of 1800 out of 2400
Partial fit of 1900 out of 2400
Partial fit of 2000 out of 2400
Partial fit of 2100 out of 2400
Partial fit of 2200 out of 2400
Partial fit of 2300 out of 2400
Partial fit of 2400 out of 2400
done in 506s
826 Chapter 5 Examples


```
scikitlearn user guide Release 0213
printdoc
import time
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets
from sklearn.cluster import MiniBatchKMeans
from sklearn.feature_extraction.image import extract_patches2d
faces = datasets.fetcholivettifaces
```

```

Learn the dictionary of images
printLearning the dictionary
rng = np.random.RandomState(0)
kmeans = MiniBatchKMeans(n_clusters=81, random_state=rng, verbose=True)
patchsize = (20, 20)
buffer = []
t0 = time.time()
    The online learning part cycle over the whole dataset 6 times
index = 0
for i in range(6):
    for i in range(len(faces.images)):
        data = extract_patches2d(img, patchsize, max_patches=50)
        random_state = rng
        data = data.reshape((data.shape[0], data.shape[1]))
        buffer.append(data)
        index = index + 1
    if index == 100:
        data = np.concatenate(buffer, axis=0)
        data = np.mean(data, axis=0)
        data = np.std(data, axis=0)
        kmeans.partial_fit(data)
        buffer = []
    if index == 100:
        printPartial fit of 4i out of i
        index = index + 1
        dt = time.time() - t0
        printdone in 2fs dt
```

```

Plot the results
plt.figure(figsize=(4, 4))
for i, patch in enumerate(kmeans.cluster_centers_):
    plt.subplot(9, 9, i + 1)
    plt.imshow(patch.reshape(patchsize), cmap=plt.cm.gray,
               interpolation='nearest')
plt.xticks()
56 Clustering 827
```

scikitlearn user guide Release 0213

plt.yticks

plt.suptitle Patches of faces nTrain time 1fs ondpatches

dt 8lenfacesimages fontsize16

plt.subplots_adjust(0.08, 0.02, 0.92, 0.85, 0.08, 0.23)

plt.show

Total running time of the script 0 minutes 6148 seconds

Note Click here to download the full example code

565 Vector Quantization Example

Face a 1024 x 768 size image of a raccoon face is used here to illustrate how kmeans is used for vector quantization

•

•

828 Chapter 5 Examples

scikitlearn user guide Release 0213

-
-

```
printdoc
Code source Gaël Varoquaux
Modified for documentation by Jaques Grobler
License BSD (Binary Search Decision) 3 clause
import numpy as np
import scipy as sp
import matplotlib.pyplot as plt
from sklearn import cluster
try SciPy 016 have face in misc
from scipymisc import face
face = facegrayTrue
except ImportError
face = sp.facegrayTrue
nclusters = 5
np.random.seed(0)
X = face.reshape(1, 1) # We need an nsample nfeature array
kmeans = cluster.KMeans(nclusters=nclusters, ninit=4)
kmeans.fit(X)
values = kmeans.cluster_centers_.squeeze()
labels = kmeans.labels_
56 Clustering 829
```

scikitlearn user guide Release 0213

create an array from labels and values

facecompressed npchooselabels values

facecompressedshape faceshape

vmin facemin

vmax facemax

original face

pltfigure1 figsize3 22

pltimshowface cmappltcmgray vminvmin vmax256

compressed face

pltfigure2 figsize3 22

pltimshowfacecompressed cmappltcmgray vminvmin vmaxvmax

equal bins face

regularvalues nplinspace0 256 nclusters 1

regularlabels npsearchsortedregularvalues face 1

regularvalues 5 regularvalues1 regularvalues1 mean

regularface npchooseregularlabelsravel regularvalues modeclip

regularfaceshape faceshape

pltfigure3 figsize3 22

pltimshowregularface cmappltcmgray vminvmin vmaxvmax

histogram

pltfigure4 figsize3 22

pltclf

pltaxes01 01 98 98

plthistX bins256 color5 edgecolor5

plt.xticks

plt.xticksregularvalues

values np.sortvalues

forcenter1 center2 inzipvalues1 values1

pltaxvline5 center1 center2 colorb

forcenter1 center2 inzipregularvalues1 regularvalues1

pltaxvline5 center1 center2 colorb linestyle

pltshow

Total running time of the script 0 minutes 3752 seconds

Note Click here to download the full example code

566 Agglomerative clustering with and without structure

This example shows the effect of imposing a connectivity graph to capture local structure in the data The graph is

simply the graph of 20 nearest neighbors

Two consequences of imposing a connectivity can be seen First clustering with a connectivity matrix is much faster

Second when using a connectivity matrix single average and complete linkage are unstable and tend to create a few

clusters that grow very quickly Indeed average and complete linkage fight this percolation behavior by considering all

the distances between two clusters when merging them while single linkage exaggerates the behaviour by considering

830 Chapter 5 Examples

scikitlearn user guide Release 0213

only the shortest distance between clusters The connectivity graph breaks this mechanism for average and complete linkage making them resemble the more brittle single linkage This effect is more pronounced for very sparse graphs try decreasing the number of neighbors in kneighborsgraph and with complete linkage In particular having a very small number of neighbors in the graph imposes a geometry that is close to that of single linkage which is well known to have this percolation instability

-
-
-

scikitlearn user guide Release 0213

```
•
  Authors Gael Varoquaux Nelle Varoquaux
  License BSD (Binary Search Decision) 3 clause
import time
import matplotlib.pyplot as plt
import numpy as np
from sklearncluster import AgglomerativeClustering
from sklearnneighbors import kneighborsgraph
  Generate sample data
nsamples 1500
nprandomseed0
t 15 np.pi1 3nprandomrand1 nsamples
x tnp.cost
y tnpsint
X np.concatenate(x, y)
X 7 nprandomrandn2 nsamples
X XT
  Create a graph capturing local connectivity Larger number of neighbors
  will give more homogeneous clusters to the cost of computation
  time A very large number of neighbors gives more evenly distributed
  cluster sizes but may not impose the local manifold structure of
  the data
knngraph kneighborsgraph(X, 30, include_self=False)
for connectivity in [None, knngraph]:
  for nclusters in [30, 3]:
    plt.figure(figsize=(10, 4))
    for index, linkage in enumerate(average, complete, ward, single):
      plt.subplot(1, 4, index + 1)
      model = AgglomerativeClustering(linkage=linkage,
        connectivity=connectivity,
        n_clusters=nclusters)
      t0 = time.time()
832 Chapter 5 Examples
```

scikitlearn user guide Release 0213
modelfitX
elapsedtime timetime t0
pltscatterX 0 X 1 cmodellabels
cmappltcmnipy spectral
plttitlelinkage sntime2fs linkage elapsedtime
fontdictdictverticalalignmenttop
pltaxisequal
pltaxisoff
pltsubplotsadjustbottom0 top89 wspace0
left0 right1
pltsubtitlencluster i connectivity r
nclusters connectivity is notNone size17
pltshow
Total running time of the script 0 minutes 1743 seconds
Note Click here to download the full example code
567 Demo of affinity propagation clustering algorithm
Reference Brendan J Frey and Delbert Dueck “Clustering by Passing Messages Between Data Points” Science Feb
2007
56 Clustering 833

scikitlearn user guide Release 0213
Out
Estimated number of clusters 3
Homogeneity 0872
Completeness 0872
Vmeasure 0872
Adjusted Rand Index 0912
Adjusted Mutual Information 0871
Silhouette Coefficient 0753
printdoc
from sklearncluster import AffinityPropagation
from sklearn import metrics
from sklearndatasetssamplesgenerator import makeblobs

Generate sample data
centers 1 1 1 1 1
X labelstrue makeblobsnsamples300 centerscenters clusterstd05
randomstate0
834 Chapter 5 Examples


```
Compute Affinity Propagation
af AffinityPropagationpreference50fitX
clustercentersindices afclustercentersindices
labels aflabels
nclusters lenclustercentersindices
printEstimated number of clusters d nclusters
printHomogeneity 03f metricshomogeneityscorelabelstrue labels
printCompleteness 03f metricscompletenessscorelabelstrue labels
printVmeasure 03f metricsvmeasurescorelabelstrue labels
printAdjusted Rand Index 03f
metricsadjustedrandscorelabelstrue labels
printAdjusted Mutual Information 03f
metricsadjustedmutualinfoscorelabelstrue labels
averagemethodarithmetic
printSilhouette Coefficient 03f
metricssilhouettescoreX labels metricsqeuclidean
```

```
Plot result
import matplotlib.pyplot as plt
from itertools import cycle
pltcloseall
pltfigure1
pltclf
colors cyclebgrcmykbgcmykbgcmykbgcmyk
fork colinziprangencusters colors
classmembers labels k
clustercenter Xclustercentersindicesk
pltplotXclassmembers 0 Xclassmembers 1 col
pltplotclustercenter0 clustercenter1 o markerfacecolorcol
markeredgecolorl markersize14
forxinXclassmembers
pltplotclustercenter0 x0 clustercenter1 x1 col
plttitleEstimated number of clusters d nclusters
pltshow
```

Total running time of the script 0 minutes 0569 seconds

Note Click here to download the full example code

568 Segmenting the picture of greek coins in regions

This example uses Spectral clustering on a graph created from voxeltovoxel difference on an image to break this image into multiple partlyhomogeneous regions

This procedure spectral clustering on an image is an efficient approximate solution for finding normalized graph cuts There are two options to assign labels

56 Clustering 835

scikitlearn user guide Release 0213

- with ‘kmeans’ spectral clustering will cluster samples in the embedding space using a kmeans algorithm
- whereas ‘discrete’ will iteratively search for the closest partition space to the embedding space

printdoc

Author Gael Varoquaux gaelvaroquauxnormalesuporg Brian Cheung

License BSD (Binary Search Decision) 3 clause

import time

import numpy as np

from distutils.version import LooseVersion

from scipy.ndimage.filters import gaussianfilter

import matplotlib.pyplot as plt

import skimage

from skimage.data import coins

from skimage.transform import rescale

from sklearn.feature_extraction import image

from sklearn.cluster import spectral_clustering

these were introduced in skimage014

if LooseVersion(skimage.version) < 0.14

rescale_params = dict(antialiasing=False, multichannel=False)

else

rescale_params

load the coins as a numpy array

orig_coins = coins

Resize it to 20 of the original size to speed up the processing

Applying a Gaussian filter for smoothing prior to downscaling

reduces aliasing artifacts

smoothed_coins = gaussianfilter(orig_coins, sigma=2)

rescaled_coins = rescale(smoothed_coins, 0.2, mode='reflect')

rescale_params

Convert the image into a graph with the value of the gradient on the edges

graph = imageimg2graph(rescaled_coins)

Take a decreasing function of the gradient an exponential

The smaller beta is the more independent the segmentation is of the

actual image For beta=1 the segmentation is close to a voronoi

beta = 10

eps = 1e6

graph_data = np.exp(beta * graph.data) graph_data_std = eps

Apply spectral clustering this step goes much faster if you have pyamg installed

N_REGIONS = 25

Visualize the resulting regions

for assign_labels in kmeans_discretize

t0 = time.time

labels = spectral_clustering(graph, n_clusters=N_REGIONS)

assign_labels, assign_labels, random_state=42

836 Chapter 5 Examples

```
scikitlearn user guide Release 0213
t1 = timetime
labels = labels.reshape(rescaledcoins.shape)
plt.figure(figsize=(5, 5))
plt.imshow(rescaledcoins, cmap=plt.cm.gray)
for i in range(NREGIONS):
    plt.contour(labels[i],
                colors=plt.cm.nipy_spectral(float(NREGIONS)
    plt.xticks(
    plt.yticks(
    title = 'Spectral clustering s2fs'
    assign_labels(t1, t0)
    print(title)
    plt.title(title)
    plt.show()
    •
56 Clustering 837
```

scikitlearn user guide Release 0213

•

Out

Spectral clustering kmeans 515s

Spectral clustering discretize 598s

Total running time of the script 0 minutes 11961 seconds

Note [Click here to download the full example code](#)

569 Kmeans Clustering

The plots display firstly what a Kmeans algorithm would yield using three clusters It is then shown what the effect of a bad initialization is on the classification process By setting ninit to only 1 default is 10 the amount of times that the algorithm will be run with different centroid seeds is reduced The next plot displays what using eight clusters would deliver and finally the ground truth

838 Chapter 5 Examples

scikitlearn user guide Release 0213

- -
- 56 Clustering 839

scikitlearn user guide Release 0213

-
-

printdoc

Code source Gaël Varoquaux

Modified for documentation by Jaques Grobler

License BSD (Binary Search Decision) 3 clause

import numpy as np

import matplotlib.pyplot as plt

Though the following import is not directly being used it is required
for 3D projection to work

from mpltoolkitsmplot3d import Axes3D

from sklearncluster import KMeans

from sklearn import datasets

nprandomseed5

840 Chapter 5 Examples

```
scikitlearn user guide Release 0213
iris datasetsloadiris
X irisdata
y iristarget
estimators kmeansiris8 KMeansnclusters8
kmeansiris3 KMeansnclusters3
kmeansirisbadinit KMeansnclusters3 ninit1
initrandom
fignum 1
titles 8 clusters 3 clusters 3 clusters bad initialization
forname est inestimators
fig pltfigurefignum figsize4 3
ax Axes3Dfig rect0 0 95 1 elev48 azim134
estfitX
labels estlabels
axscatterX 3 X 0 X 2
clabelsastypenpfloat edgecolork
axwxaxissetticklabels
axwyaxissetticklabels
axwzaxissetticklabels
axsetxlabelPetal width
axsetylabelSepal length
axsetzlabelPetal length
axsettitletitlesfignum 1
axdist 12
fignum fignum 1
Plot the ground truth
fig pltfigurefignum figsize4 3
ax Axes3Dfig rect0 0 95 1 elev48 azim134
forname label inSetosa 0
Versicolour 1
Virginica 2
axtext3DXy label 3mean
Xy label 0mean
Xy label 2mean 2 name
horizontalalignmentcenter
bboxdictalpha2 edgecolorw facecolorw
Reorder the labels to have colors matching the cluster results
y npchoosey 1 2 0astypenpfloat
axscatterX 3 X 0 X 2 cy edgecolork
axwxaxissetticklabels
axwyaxissetticklabels
axwzaxissetticklabels
axsetxlabelPetal width
axsetylabelSepal length
axsetzlabelPetal length
axsettitleGround Truth
axdist 12
figshow
Total running time of the script 0 minutes 0241 seconds
56 Clustering 841
```

scikitlearn user guide Release 0213

Note [Click here to download the full example code](#)

5610 Various Agglomerative Clustering on a 2D embedding of digits

An illustration of various linkage option for agglomerative clustering on a 2D embedding of the digits dataset

The goal of this example is to show intuitively how the metrics behave and not to find good clusters for the digits

This is why the example works on a 2D embedding

What this example shows us is the behavior “rich getting richer” of agglomerative clustering that tends to create uneven cluster sizes This behavior is pronounced for the average linkage strategy that ends up with a couple of singleton clusters while in the case of single linkage we get a single central cluster with all other clusters being drawn from noise points around the fringes

-

842 Chapter 5 Examples

scikitlearn user guide Release 0213

- -
- 56 Clustering 843

scikitlearn user guide Release 0213

•

Out

Computing embedding

Done

ward 046s

average 036s

complete 038s

single 017s

Authors Gael Varoquaux

License BSD (Binary Search Decision) 3 clause C INRIA 2014

printdoc

from time import time

import numpy as np

from scipy import ndimage

from matplotlib import pyplotasplt

from sklearn import manifold datasets

digits datasetsloadaddigitsnclass10

X digitsdata

y digitstarget

nsamples nfeatures Xshape

844 Chapter 5 Examples

```
scikitlearn user guide Release 0213
nprandomseed0
defnudgeimagesX y
    Having a larger dataset shows more clearly the behavior of the
    methods but we multiply the size of the dataset only by 2 as the
    cost of the hierarchical clustering methods are strongly
    superlinear in nsamples
    shift lambdax ndimageshiftxreshape8 8
    3nprandomnormalsize2
    modeconstant
    ravel
    X npconcatenateX npapplyalongaxisshift 1 X
    Y npconcatenatey y axis0
    returnX Y
X y nudgeimagesX y
```

```
Visualize the clustering
defplotclusteringXred labels titleNone
xmin xmax npminXred axis0 npmaxXred axis0
Xred Xred xmin xmax xmin
pltfigurefigsize6 4
fori inrangeXredshape0
    plttextXredi 0 Xredi 1 stryi
    colorpltcmnipy spectrallabels i 10
    fontdictweight bold size 9
    pltxticks
    pltyticks
    iftitleis notNone
        plttitletitle size17
    pltaxisoff
    plttightlayoutrect0 003 1 095
```

```
2D embedding of the digits dataset
printComputing embedding
Xred manifoldSpectralEmbeddingncomponents2fittransformX
printDone
from sklearncluster import AgglomerativeClustering
forlinkage inward average complete single
clustering AgglomerativeClusteringlinkagelinkage nclusters10
t0 time
clusteringfitXred
printst2fs linkage time t0
plotclusteringXred clusteringlabels slinkage linkage
pltshow
Total running time of the script 0 minutes 26873 seconds
56 Clustering 845
```

scikitlearn user guide Release 0213

Note [Click here to download the full example code](#)

5611 Spectral clustering for image segmentation

In this example an image with connected circles is generated and spectral clustering is used to separate the circles In these settings the Spectral clustering approach solves the problem know as ‘normalized graph cuts’ the image is seen as a graph of connected voxels and the spectral clustering algorithm amounts to choosing graph cuts defining regions while minimizing the ratio of the gradient along the cut and the volume of the region

As the algorithm tries to balance the volume ie balance the region sizes if we take circles with different sizes the segmentation fails

In addition as there is no useful information in the intensity of the image or its gradient we choose to perform the spectral clustering on a graph that is only weakly informed by the gradient This is close to performing a V oronoi partition of the graph

In addition we use the mask of the objects to restrict the graph to the outline of the objects In this example we are interested in separating the objects one from the other and not from the background

•

scikitlearn user guide Release 0213

- 56 Clustering 847

scikitlearn user guide Release 0213

```
•
printdoc
Authors Emmanuelle Guillard emmanuelleguillardnormalesuporg
Gael Varoquaux gaelvaroquauxnormalesuporg
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearnfeatureextraction import image
from sklearncluster import spectralclustering
l = 100
x, y = np.indices(l)
center1 = 28, 24
center2 = 40, 50
center3 = 67, 58
center4 = 24, 70
radius1, radius2, radius3, radius4 = 16, 14, 15, 14
circle1 = x - center1[0], y - center1[1], radius1
circle2 = x - center2[0], y - center2[1], radius2
circle3 = x - center3[0], y - center3[1], radius3
circle4 = x - center4[0], y - center4[1], radius4
56 Clustering 849
```

```
4 circles
img circle1 circle2 circle3 circle4
We use a mask that limits to the foreground the problem that we are
interested in here is not separating the objects from the background
but separating them one from the other
mask imgastypebool
img imgastypefloat
img 1 02 nprandomrandn imgshape
Convert the image into a graph with the value of the gradient on the
edges
graph imageimgtographimg maskmask
Take a decreasing function of the gradient we take it weakly
dependent from the gradient the segmentation is close to a voronoi
graphdata npexpgraphdata graphdatastd
Force the solver to be arpack since amg is numerically
unstable on this example
labels spectralclusteringgraph nclusters4 eigensolverarpack
labelim npfullmaskshape 1
labelimmask labels
pltmatshowimg
pltmatshowlabelim
```

```
2 circles
img circle1 circle2
mask imgastypebool
img imgastypefloat
img 1 02 nprandomrandn imgshape
graph imageimgtographimg maskmask
graphdata npexpgraphdata graphdatastd
labels spectralclusteringgraph nclusters2 eigensolverarpack
labelim npfullmaskshape 1
labelimmask labels
pltmatshowimg
pltmatshowlabelim
pltshow
Total running time of the script 0 minutes 0675 seconds
Note Click here to download the full example code
850 Chapter 5 Examples
```


scikitlearn user guide Release 0213

5612 A demo of structured Ward hierarchical clustering on an image of coins

Compute the segmentation of a 2D image with Ward hierarchical clustering The clustering is spatially constrained in order for each segmented region to be in one piece

Out

Compute structured hierarchical clustering

Elapsed time 02273859977722168

Number of pixels 4697

Number of clusters 27

Author Vincent Michel 2010

Alexandre Gramfort 2011

License BSD (Binary Search Decision) 3 clause

printdoc

import time as time

56 Clustering 851

```
scikitlearn user guide Release 0213
import numpy as np
from distutils.version import LooseVersion
from scipy.ndimage.filters import gaussian_filter
import matplotlib.pyplot as plt
import skimage
from skimage.data import coins
from skimage.transform import rescale
from sklearn.feature_extraction.image import grid_to_graph
from sklearn.cluster import AgglomerativeClustering
# these were introduced in skimage 0.14
if LooseVersion(skimage.__version__) < LooseVersion('0.14'):
    rescale_params = dict(anti_aliasing=False, multichannel=False)
else:
    rescale_params = {}

# Generate data
orig_coins = coins
# Resize it to 20% of the original size to speed up the processing
# Applying a Gaussian filter for smoothing prior to downscaling
# reduces aliasing artifacts
smoothed_coins = gaussian_filter(orig_coins, sigma=2)
rescaled_coins = rescale(smoothed_coins, 0.2, mode='reflect',
                        **rescale_params)
X = np.reshape(rescaled_coins, (-1, 1))

# Define the structure A of the data: Pixels connected to their neighbors
connectivity = grid_to_graph(rescaled_coins.shape)

# Compute clustering
print('Compute structured hierarchical clustering')
st = time.time()
n_clusters = 27 # number of regions
ward = AgglomerativeClustering(n_clusters=n_clusters, linkage='ward',
                             connectivity=connectivity)
ward.fit(X)
label = np.reshape(ward.labels_, rescaled_coins.shape)
print('Elapsed time: %s' % (time.time() - st))
print('Number of pixels: %s' % label.size)
print('Number of clusters: %s' % np.unique(label).size)

# Plot the results on an image
plt.figure(figsize=(5, 5))
plt.imshow(rescaled_coins, cmap=plt.cm.gray)
for i in range(n_clusters):
    plt.contour(label == i, colors=plt.cm.nipy_spectral(float(i)/n_clusters))
plt.xticks()
plt.yticks()
852 Chapter 5 Examples
```

scikitlearn user guide Release 0213
pltshow
Total running time of the script 0 minutes 0897 seconds
Note Click here to download the full example code
5613 Demo of DBSCAN clustering algorithm
Finds core samples of high density and expands clusters from them
Out
Estimated number of clusters 3
Estimated number of noise points 18
Homogeneity 0953
Completeness 0883
Vmeasure 0917
Adjusted Rand Index 0952
Adjusted Mutual Information 0916
Silhouette Coefficient 0626
56 Clustering 853

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
from sklearncluster import DBSCAN
from sklearn import metrics
from sklearndatasetssamplesgenerator import makeblobs
from sklearnpreprocessing import StandardScaler

Generate sample data
centers 1 1 1 1 1 1
X labelstrue makeblobsnsamples750 centerscenters clusterstd04
randomstate0
X StandardScalerfittransformX
```

```
Compute DBSCAN
db DBSCANeps03 minsamples10fitX
coresamplesmask npzeroslikedblabels dtypebool
coresamplesmaskdbcoresampleindices True
labels dblabels
Number of clusters in labels ignoring noise if present
nclusters lensetlabels 1 if1inlabelselse0
nnoise listlabelscount1
printEstimated number of clusters d nclusters
printEstimated number of noise points d nnoise
printHomogeneity 03f metricshomogeneityscorelabelstrue labels
printCompleteness 03f metricscompletenessscorelabelstrue labels
printVmeasure 03f metricsvmeasurescorelabelstrue labels
printAdjusted Rand Index 03f
metricsadjustedrandscorelabelstrue labels
printAdjusted Mutual Information 03f
metricsadjustedmutualinfoscorelabelstrue labels
averagemethodarithmetic
printSilhouette Coefficient 03f
metricssilhouettescoreX labels
```

```
Plot result
import matplotlib.pyplot as plt
Black removed and is used for noise instead
unique_labels setlabels
colors pltcmSpectraleach
foreachinnplinspace0 1 lenunique_labels
fork colinzipunique_labels colors
ifk 1
Black used for noise
col 0 0 0 1
854 Chapter 5 Examples
```

scikitlearn user guide Release 0213

classmembermask labels k

xy Xclassmembermask coresamplesmask

pltplotxy 0 xy 1 o markerfacecolortuplecol

markeredgecolork markersize14

xy Xclassmembermask coresamplesmask

pltplotxy 0 xy 1 o markerfacecolortuplecol

markeredgecolork markersize6

plttitleEstimated number of clusters d nclusters

pltshow

Total running time of the script 0 minutes 0043 seconds

Note Click here to download the full example code

5614 Color Quantization using KMeans

Performs a pixelwise Vector Quantization VQ of an image of the summer palace China reducing the number of colors required to show the image from 96615 unique colors to 64 while preserving the overall appearance quality In this example pixels are represented in a 3Dspace and Kmeans is used to find 64 color clusters In the image processing literature the codebook obtained from Kmeans the cluster centers is called the color palette Using a single byte up to 256 colors can be addressed whereas an RGB encoding requires 3 bytes per pixel The GIF file format for example uses such a palette For comparison a quantized image using a random codebook colors picked up randomly is also shown

56 Clustering 855

scikitlearn user guide Release 0213

- Out
Fitting model on a small subsample of the data
done in 0264s
Predicting color indices on the full image kmeans
done in 0216s
Predicting color indices on the full image random
done in 0310s
Authors Robert Layton robertlaytongmailcom
Olivier Grisel oliviergriselenstaorg
Mathieu Blondel mathieumblondelorg

License BSD (Binary Search Decision) 3 clause
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearncluster import KMeans
from sklearnmetrics import pairwisedistancesargmin
from sklearndatasets import loadsampleimage
from sklearnutils import shuffle

858 Chapter 5 Examples


```
scikitlearn user guide Release 0213
from time import time
ncolors = 64
    Load the Summer Palace photo
china = loadsampleimagechinajpg
    Convert to floats instead of the default 8 bits integer coding Dividing by
    255 is important so that pltimshow behaves works well on float data need to
    be in the range 01
china = nparraychina dtype=npfloat64 / 255
    Load Image and transform to a 2D numpy array
w, h, d = originalshape = tuple(chinashape)
assert d == 3
imagearray = npreshape(china, (w, h, d))
printFitting model on a small subsample of the data
t0 = time
imagearraysample = shuffleimagearray(imagearray, randomstate=0)1000
kmeans = KMeans(nclusters=ncolors, randomstate=0)fitimagearraysample
printdone in %3fs %time - t0
    Get labels for all points
printPredicting color indices on the full image kmeans
t0 = time
labels = kmeanspredictimagearray(imagearray)
printdone in %3fs %time - t0
codebookrandom = shuffleimagearray(imagearray, randomstate=0)ncolors
printPredicting color indices on the full image random
t0 = time
labelsrandom = pairwise_distances(imagearray, codebookrandom)
imagearray = imagearray - labelsrandom
axis0 = imagearray[:, 0]
printdone in %3fs %time - t0
defrecreateimagecodebook(labels, w, h):
    Recreate the compressed image from the code book labels
    d = codebookshape[1]
    image = np.zeros((w, h, d))
    labelidx = 0
    for i in range(w):
        for j in range(h):
            image[i, j, :] = codebook[labels[labelidx], :]
            labelidx += 1
    return image
    Display all results alongside original image
pltfigure1 = plt.imshow(china)
pltclf()
pltaxisoff()
plttitleOriginal image 96615 colors
pltimshowchina
pltfigure2 = plt.imshow(imagearray)
56 Clustering 859
```

scikitlearn user guide Release 0213

```
pltclf
pltaxisoff
plttitleQuantized image 64 colors KMeans
pltimshowrecreateimagekmeansclustercenters labels w h
pltfigure3
pltclf
pltaxisoff
plttitleQuantized image 64 colors Random
pltimshowrecreateimagecodebookrandom labelsrandom w h
pltshow
```

Total running time of the script 0 minutes 1400 seconds

Note Click [here](#) to download the full example code

5615 Hierarchical clustering structured vs unstructured ward

Example builds a swiss roll dataset and runs hierarchical clustering on their position

For more information see Hierarchical clustering

In a first step the hierarchical clustering is performed without connectivity constraints on the structure and is solely based on distance whereas in a second step the clustering is restricted to the kNearest Neighbors graph it's a hierarchical clustering with structure prior

Some of the clusters learned without connectivity constraints do not respect the structure of the swiss roll and extend across different folds of the manifolds On the opposite when opposing connectivity constraints the clusters form a nice parcellation of the swiss roll

860 Chapter 5 Examples

scikitlearn user guide Release 0213

•

Out

Compute unstructured hierarchical clustering

Elapsed time 008s

Number of points 1500

Compute structured hierarchical clustering

Elapsed time 007s

Number of points 1500

Authors Vincent Michel 2010

Alexandre Gramfort 2010

Gael Varoquaux 2010

License BSD (Binary Search Decision) 3 clause

printdoc

import time as time

import numpy as np

import matplotlib.pyplot as plt

import mpltoolkitsmplot3daxes3d as p3

from sklearncluster import AgglomerativeClustering

from sklearndatasetssamplesgenerator import makeswissroll

862 Chapter 5 Examples

scikitlearn user guide Release 0213

```
Generate data swiss roll dataset
nsamples 1500
noise 005
X makeswissrollnsamples noise
Make it thinner
X 1 5
```

```
Compute clustering
printCompute unstructured hierarchical clustering
st timetime
ward AgglomerativeClusteringnclusters6 linkagewardfitX
elapsedtime timetime st
label wardlabels
printElapsed time 2fs elapsedtime
printNumber of points i labelsizesize
```

```
Plot result
fig pltfigure
ax p3Axes3Dfig
axviewinit7 80
forlinnpuniqueylabel
axscatterXlabel 1 0 Xlabel 1 1 Xlabel 1 2
colorpltcmjetnpfloatl npmaxlabel 1
s20 edgecolork
plttitleWithout connectivity constraints time 2fs elapsedtime
```

```
Define the structure A of the data Here a 10 nearest neighbors
from sklearnneighbors import kneighborsgraph
connectivity kneighborsgraphX nneighbors10 includeselfFalse
```

```
Compute clustering
printCompute structured hierarchical clustering
st timetime
ward AgglomerativeClusteringnclusters6 connectivityconnectivity
linkagewardfitX
elapsedtime timetime st
label wardlabels
printElapsed time 2fs elapsedtime
printNumber of points i labelsizesize
```

```
Plot result
fig pltfigure
ax p3Axes3Dfig
axviewinit7 80
forlinnpuniqueylabel
axscatterXlabel 1 0 Xlabel 1 1 Xlabel 1 2
colorpltcmjetfloatl npmaxlabel 1
s20 edgecolork
plttitleWith connectivity constraints time 2fs elapsedtime
56 Clustering 863
```

scikitlearn user guide Release 0213

pltshow

Total running time of the script 0 minutes 0202 seconds

Note Click here to download the full example code

5616 Agglomerative clustering with different metrics

Demonstrates the effect of different metrics on the hierarchical clustering

The example is engineered to show the effect of the choice of different metrics It is applied to waveforms which can be seen as highdimensional vector Indeed the difference between metrics is usually more pronounced in high dimension in particular for euclidean and cityblock

We generate data from three groups of waveforms Two of the waveforms waveform 1 and waveform 2 are proportional one to the other The cosine distance is invariant to a scaling of the data as a result it cannot distinguish these two waveforms Thus even with no noise clustering using this distance will not separate out waveform 1 and 2 We add observation noise to these waveforms We generate very sparse noise only 6 of the time points contain noise As a result the l1 norm of this noise ie “cityblock” distance is much smaller than it’s l2 norm “euclidean” distance This can be seen on the interclass distance matrices the values on the diagonal that characterize the spread of the class are much bigger for the Euclidean distance than for the cityblock distance

When we apply clustering to the data we find that the clustering reflects what was in the distance matrices Indeed for the Euclidean distance the classes are illseparated because of the noise and thus the clustering does not separate the waveforms For the cityblock distance the separation is good and the waveform classes are recovered Finally the cosine distance does not separate at all waveform 1 and 2 thus the clustering puts them in the same cluster

864 Chapter 5 Examples

scikitlearn user guide Release 0213

```
•
  Author Gael Varoquaux
  License BSD (Binary Search Decision) 3Clause or CC0
import matplotlib.pyplot as plt
import numpy as np
from sklearn.cluster import AgglomerativeClustering
from sklearn.metrics import pairwise_distances
np.random.seed(0)
# Generate waveform data
nfeatures = 2000
t = np.linspace(0, 1, nfeatures)
def sqrx
return np.sign(np.cos(x))
X = list
y = list
for i, phi, a in enumerate(5, 15, 5, 6, 3, 2):
    for i in range(30):
        phase_noise = 0.1 * np.random.normal()
        amplitude_noise = 0.4 * np.random.normal()
        additional_noise = 1.2 * np.random.randn(nfeatures)
        # Make the noise sparse
56 Clustering 871
```

```
scikitlearn user guide Release 0213
additionalnoisenpabsadditionalnoise 997 0
Xappend12 a amplitudenoise
sqr6t phi phasenoise
additionalnoise
yappendi
X nparrayX
y nparrayy
nclusters 3
labels Waveform 1 Waveform 2 Waveform 3
Plot the groundtruth labelling
pltfigure
pltaxes0 0 1 1
forl c n inziprangencusters rgb
labels
lines pltplotXy IT cc alpha5
lines0setlabeln
pltlegendlocbest
pltaxistight
pltaxisoff
pltsuptitleGround truth size20
Plot the distances
forindex metric inenumeratecosine euclidean cityblock
avgdist npzerosnclusters nclusters
pltfigurefigsize5 45
foriinrangencusters
forjinnrangencusters
avgdisti j pairwisedistancesXy i Xy j
metricmetricmean
avgdist avgdistmax
foriinrangencusters
forjinnrangencusters
plttexti j 53f avgdisti j
verticalalignmentcenter
horizontalalignmentcenter
pltimshowavgdist interpolationnearest cmappltcmgnuplot2
vmin0
pltxticksrangencusters labels rotation45
plttyicksrangencusters labels
pltcolorbar
pltsuptitleInterclass sdistances metric size18
plttightlayout
Plot clustering results
forindex metric inenumeratecosine euclidean cityblock
model AgglomerativeClusteringnclustersnclusters
linkageaverage affinitymetric
modelfitX
872 Chapter 5 Examples
```

scikitlearn user guide Release 0213

```
pltfigure
pltaxes0 0 1 1
forl cinzipnparangemodelInclusters rgbk
pltplotXmodellabels IT cc alpha5
pltaxistight
pltaxisoff
pltstuptitleAgglomerativeClusteringaffinity s metric size20
pltshow
```

Total running time of the script 0 minutes 0584 seconds

Note [Click here to download the full example code](#)

5617 Inductive Clustering

Clustering can be expensive especially when our dataset contains millions of datapoints Many clustering algorithms are not inductive and so cannot be directly applied to new data samples without recomputing the clustering which may be intractable Instead we can use clustering to then learn an inductive model with a classifier which has several benefits

- it allows the clusters to scale and apply to new data
- unlike refitting the clusters to new samples it makes sure the labelling procedure is consistent over time
- it allows us to use the inferential capabilities of the classifier to describe or explain the clusters

This example illustrates a generic implementation of a metaestimator which extends clustering by inducing a classifier from the cluster labels

Authors Chirag Nagpal

Christos Aridas

```
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearnbase import BaseEstimator clone
from sklearncluster import AgglomerativeClustering
from sklearndatasets import makeblobs
from sklearnensemble import RandomForestClassifier
56 Clustering 873
```

```
scikitlearn user guide Release 0213
from sklearnutilsmetaestimators import ifdelegatehasmethod
NSAMPLES 5000
RANDOMSTATE 42
class InductiveClusterer BaseEstimator
definitself clusterer classifier
selfclusterer clusterer
selfclassifier classifier
defitself X yNone
selfclusterer cloneselfclusterer
selfclassifier cloneselfclassifier
y selfclustererfitpredictX
selfclassifierfitX y
returnself
ifdelegatehasmethod delegateclassifier
defpredictself X
returnselfclassifierpredictX
ifdelegatehasmethod delegateclassifier
defdecisionfunctionself X
returnselfclassifierdecisionfunctionX
defplotsscatterX color alpha05
returnpltscatterX 0
X 1
ccolor
alphaalpha
edgecolork
Generate some training data from clustering
X y makeblobsnsamplesNSAMPLES
clusterstd10 10 05
centers5 5 0 0 5 5
randomstateRANDOMSTATE
Train a clustering algorithm on the training data and get the cluster labels
clusterer AgglomerativeClusteringnclusters3
clusterlabels clustererfitpredictX
pltfigurefigsize12 4
pltsubplot131
plotsscatterX clusterlabels
plttitleWard Linkage
Generate new samples and plot them along with the original dataset
Xnew ynew makeblobsnsamples10
centers7 1 2 4 3 6
randomstateRANDOMSTATE
874 Chapter 5 Examples
```


scikitlearn user guide Release 0213

pltsubplot132

plotscatterX clusterlabels

plotscatterXnew black 1

plttitleUnknown instances

Declare the inductive learning model that it will be used to

predict cluster membership for unknown instances

classifier RandomForestClassifierrandomstateRANDOMSTATE

inductivelearner InductiveClustererclusterer classifierfitX

probableclusters inductivelearnerpredictXnew

pltsubplot133

plotscatterX clusterlabels

plotscatterXnew probableclusters

Plotting decision regions

xmin xmax X 0min 1 X 0max 1

ymin ymax X 1min 1 X 1max 1

xx yy npmeshgridnparangexmin xmax 01

nparangeymin ymax 01

Z inductivelearnerpredictnpcxxravel yy.ravel

Z Z.reshapexxshape

pltcontourfxx yy Z alpha04

plttitleClassify unknown instances

pltshow

Total running time of the script 0 minutes 1436 seconds

Note Click here to download the full example code

5618 Demo of OPTICS clustering algorithm

Finds core samples of high density and expands clusters from them This example uses data that is generated so

that the clusters have different densities The sklearnclusterOPTICS is first used with its Xi cluster detec

tion method and then setting specific thresholds on the reachability which corresponds to sklearncluster

DBSCAN We can see that the different clusters of OPTICS's Xi method can be recovered with different choices of

thresholds in DBSCAN

56 Clustering 875

```
scikitlearn user guide Release 0213
Authors Shane Grigsby refugeroctaluscom
Adrin Jalali adrinjalaligmailcom
License BSD (Binary Search Decision) 3 clause
from sklearncluster import OPTICS clusteropticsdbscan
import matplotlibgridspec as gridspec
import matplotlibpyplot as plt
import numpy as np
Generate sample data
nprandomseed0
npointspercluster 250
C1 5 2 8 nprandomrandnnpointspercluster 2
C2 4 1 1 nprandomrandnnpointspercluster 2
C3 1 2 2 nprandomrandnnpointspercluster 2
C4 2 3 3 nprandomrandnnpointspercluster 2
C5 3 2 16 nprandomrandnnpointspercluster 2
C6 5 6 2 nprandomrandnnpointspercluster 2
X npvstackC1 C2 C3 C4 C5 C6
clust OPTICSminsamples50 xi05 minclustersize05
Run the fit
clustfitX
876 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
labels050 clusteropticsdbscanreachabilityclustreachability
coredistancesclustcoredistances
orderingclustordering eps05
labels200 clusteropticsdbscanreachabilityclustreachability
coredistancesclustcoredistances
orderingclustordering eps2
space nparangelenX
reachability clustreachabilityclustordering
labels clustlabelsclustordering
pltfigurefigsize10 7
G gridspecGridSpec2 3
ax1 pltsubplotG0
ax2 pltsubplotG1 0
ax3 pltsubplotG1 1
ax4 pltsubplotG1 2
Reachability plot
colors g r b y c
forklass color inziprange0 5 colors
Xk spacelabels klass
Rk reachabilitylabels klass
ax1plotXk Rk color alpha03
ax1plotspacelabels 1 reachabilitylabels 1 k alpha03
ax1plotspace npfulllikespace 2 dtypefloat k alpha05
ax1plotspace npfulllikespace 05 dtypefloat k alpha05
ax1setylabelReachability epsilon distance
ax1settitleReachability Plot
OPTICS
colors g r b y c
forklass color inziprange0 5 colors
Xk Xclustlabels klass
ax2plotXk 0 Xk 1 color alpha03
ax2plotXclustlabels 1 0 Xclustlabels 1 1 k alpha01
ax2settitleAutomatic Clustering nOPTICS
DBSCAN at 05
colors g greenyellow olive r b c
forklass color inziprange0 6 colors
Xk Xlabels050 klass
ax3plotXk 0 Xk 1 color alpha03 marker
ax3plotXlabels050 1 0 Xlabels050 1 1 k alpha01
ax3settitleClustering at 05 epsilon cut nDBSCAN
DBSCAN at 2
colors g m y c
forklass color inziprange0 4 colors
Xk Xlabels200 klass
ax4plotXk 0 Xk 1 color alpha03
ax4plotXlabels200 1 0 Xlabels200 1 1 k alpha01
ax4settitleClustering at 20 epsilon cut nDBSCAN
plttightlayout
pltshow
Total running time of the script 0 minutes 0935 seconds
56 Clustering 877
```

scikitlearn user guide Release 0213

Note Click here to download the full example code

5619 Compare BIRCH and MiniBatchKMeans

This example compares the timing of Birch with and without the global clustering step and MiniBatchKMeans on a synthetic dataset having 100000 samples and 2 features generated using makeblobs

Ifnclusters is set to None the data is reduced from 100000 samples to a set of 158 clusters This can be viewed as a preprocessing step before the final global clustering step that further reduces these 158 clusters to 100 clusters

Out
Birch without global clustering as the final step took 247 seconds

nclusters 158

Birch with global clustering as the final step took 290 seconds

nclusters 100

Time taken to run MiniBatchKMeans 360 seconds

Authors Manoj Kumar manojkumarsivaraj334gmailcom

Alexandre Gramfort alexandregamforttelecomparistechfr

License BSD (Binary Search Decision) 3 clause

printdoc

from itertools import cycle

from time import time

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.colors as colors

from sklearncluster import Birch MiniBatchKMeans

from sklearnndatasetssamplesgenerator import makeblobs

Generate centers for the blobs so that it forms a 10 X 10 grid

878 Chapter 5 Examples

```
scikitlearn user guide Release 0213
xx  nplinspace22 22 10
yy  nplinspace22 22 10
xx yy  npmeshgridxx yy
ncentres  nphstacknpravelxx npnewaxis
npravelyy npnewaxis
    Generate blobs to do a comparison between MiniBatchKMeans and Birch
X y  makeblobsnsamples100000 centersncentres randomstate0
    Use all colors that matplotlib provides by default
colors  cyclecolorscnameskeys
fig  pltfigurefigsize12 4
figsubplotsadjustleft004 right098 bottom01 top09
    Compute clustering with Birch with and without the final clustering step
    and plot
birchmodels  Birchthreshold17 nclustersNone
Birchthreshold17 nclusters100
finalstep  without global clustering with global clustering
forind birchmodel info inenumeratezipbirchmodels finalstep
t  time
birchmodelfitX
time  time t
printBirch sas the final step took 02fseconds
info time t
    Plot result
labels  birchmodellabels
centroids  birchmodelsubclustercenters
nclusters  npuniquelabelssize
printnclusters d nclusters
ax  figaddsubplot1 3 ind 1
forthiscentroid k col inzipcentroids rangenclusters colors
mask  labels k
axscatterXmask 0 Xmask 1
cw edgecolorcol marker alpha05
ifbirchmodelnclusters isNone
axscatterthiscentroid0 thiscentroid1 marker
ck s25
axsetylim25 25
axsetxlim25 25
axsetautoscaleyonFalse
axsettitleBirch s info
    Compute clustering with MiniBatchKMeans
mbk  MiniBatchKMeansinitkmeans nclusters100 batchsize100
ninit10 maxnoimprovement10 verbose0
randomstate0
t0  time
mbkfitX
tminibatch  time t0
printTime taken to run MiniBatchKMeans 02fseconds tminibatch
mbkmeanslabelsunique  npuniquembklabels
ax  figaddsubplot1 3 3
56 Clustering 879
```

```
scikitlearn user guide Release 0213
forthiscentroid k col inzipmbkclustercenters
rangencusters colors
mask mbklabels k
axscatterXmask 0 Xmask 1 marker
cw edgecolorcol alpha05
axscatterthiscentroid0 thiscentroid1 marker
ck s25
axsetxlim25 25
axsetylim25 25
axsettitleMiniBatchKMeans
axsetautoscaleyonFalse
pltshow
```

Total running time of the script 0 minutes 10943 seconds

Note [Click here to download the full example code](#)

5620 Empirical evaluation of the impact of kmeans initialization

Evaluate the ability of kmeans initializations strategies to make the algorithm convergence robust as measured by the relative standard deviation of the inertia of the clustering ie the sum of squared distances to the nearest cluster center

The first plot shows the best inertia reached for each combination of the model KMeans orMiniBatchKMeans and the init method initempty orinitkmeans for increasing values of the ninit parameter that controls the number of initializations

The second plot demonstrate one single run of the MiniBatchKMeans estimator using a initempty and ninit1 This run leads to a bad convergence local optimum with estimated centers stuck between ground truth clusters

The dataset used for evaluation is a 2D grid of isotropic Gaussian clusters widely spaced

scikitlearn user guide Release 0213

- Out

Evaluation of KMeans with kmeans init
Evaluation of KMeans with random init
Evaluation of MiniBatchKMeans with kmeans init
Evaluation of MiniBatchKMeans with random init
printdoc
Author Olivier Grisel oliviergrisenstaorg
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm
from sklearnutils import shuffle
from sklearnutils import checkrandomstate
from sklearncluster import MiniBatchKMeans
from sklearncluster import KMeans
randomstate np.random.RandomState(0)
882 Chapter 5 Examples


```
scikitlearn user guide Release 0213
Number of run with randomly generated dataset for each strategy so as
to be able to compute an estimate of the standard deviation
nruns 5
kmeans models can do several random inits so as to be able to trade
CPU time for convergence robustness
nitrangle nparray1 5 10 15 20
Datasets generation parameters
nsamplespercenter 100
gridsize 3
scale 01
nclusters gridSize 2
defmakedatarandomstate nsamplespercenter gridSize scale
randomstate checkrandomstaterandomstate
centers nparrayi j
foriinrangeGridsize
forjijnrangeGridsize
nclusterstrue nfeatures centersshape
noise randomstatenormal
scalescale sizensamplespercenter centersshape1
X npconcatenatec noise forcinceneters
y npconcatenatei nsamplespercenter
foriinrangencclusterstrue
returnshufflex y randomstaterandomstate
Part 1 Quantitative evaluation of various init methods
pltfigure
plots
legends
cases
KMeans kmeans
KMeans random
MiniBatchKMeans kmeans maxnoimprovement 3
MiniBatchKMeans random maxnoimprovement 3 initSize 500

forfactory init params incases
printEvaluation of swithsinit factoryname init
inertia npemptyLennitrangle nruns
forrunidinrangennruns
X y makedatarunid nsamplespercenter gridSize scale
fori ninit inenumeratenitrangle
km factorynclustersnclusters initinit randomstaterunid
ninitninit paramsfitX
inertiai runid kminertia
p plterrorbarnitrangle inertiaeanaaxis1 inertiastdaxis1
plotsappendp0
legendsappend swithsinit factoryname init
56 Clustering 883
```

scikitlearn user guide Release 0213

```
plt.xlabel('inertia')
plt.ylabel('inertia')
plt.legend()
plt.title('Mean inertia for various kmeans init across druns')
# Part 2 Qualitative visual inspection of the convergence
X, y = make_data_random(n_samples=100, n_features=2, scale=1)
km = MiniBatchKMeans(n_clusters=5, init='random', n_init=10)
random_state = random_state_fit(X)
plt.figure()
for k in range(1, 5):
    km = MiniBatchKMeans(n_clusters=k, init='random', n_init=10)
    km.fit(X)
    plt.plot(km.cluster_centers_)
    plt.scatter(X, y, c=y, marker='o')
    plt.title('Example cluster allocation with a single random init')
plt.show()
```

Total running time of the script: 0 minutes 26.95 seconds

Note: Click [here](#) to download the full example code

5621 Adjustment for chance in clustering performance evaluation

The following plots demonstrate the impact of the number of clusters and number of samples on various clustering performance evaluation metrics

Nonadjusted measures such as the VMeasure show a dependency between the number of clusters and the number of samples: the mean VMeasure of random labeling increases significantly as the number of clusters is closer to the total number of samples used to compute the measure

Adjusted for chance measure such as ARI display some random variations centered around a mean score of 0.0 for any number of samples and clusters

Only adjusted measures can hence safely be used as a consensus index to evaluate the average stability of clustering algorithms for a given value of k on various overlapping subsamples of the dataset

884 Chapter 5 Examples

scikitlearn user guide Release 0213

•

Out

Computing adjustedrandscore for 10 values of nclusters and nsamples100
done in 0026s

Computing vmeasurescore for 10 values of nclusters and nsamples100
done in 0040s

Computing amiscore for 10 values of nclusters and nsamples100
done in 0349s

Computing mutualinfoscore for 10 values of nclusters and nsamples100
done in 0033s

Computing adjustedrandscore for 10 values of nclusters and nsamples1000
done in 0041s

Computing vmeasurescore for 10 values of nclusters and nsamples1000
done in 0053s

Computing amiscore for 10 values of nclusters and nsamples1000
done in 0208s

Computing mutualinfoscore for 10 values of nclusters and nsamples1000
done in 0043s

printdoc

Author Olivier Grisel oliviergrisenstaorg

886 Chapter 5 Examples

scikitlearn user guide Release 0213
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from time import time
from sklearn import metrics
def uniform_labeling_score(scorefunc, nsamples, nclustersrange, fixednclasses=None, nruns=5, seed=42):
 Compute score for 2 random uniform cluster labelings
 Both random labelings have the same number of clusters for each value
 possible value in nclustersrange
 When fixednclasses is not None the first labeling is considered a ground
 truth class assignment with fixed number of classes

```
random_labels = np.random.RandomState(seed).randint(
    scores = np.zeros(len(nclustersrange), nruns)
    if fixednclasses is not None:
        labels_a = random_labels[low0:high, fixednclasses: sizesamples]
        for i in enumerate(nclustersrange):
            for j in range(nruns):
                if fixednclasses is None:
                    labels_a = random_labels[low0:high, k: sizesamples]
                    labels_b = random_labels[low0:high, k: sizesamples]
                    scores[i, j] = scorefunc(labels_a, labels_b)
    return scores
def ami_score(U, V):
    return metrics.adjusted_mutual_info_score(U, V)
average_method = arithmetic
score_funcs = [
    metrics.adjusted_rand_score,
    metrics.v_measure_score,
    ami_score,
    metrics.mutual_info_score]
```

2 independent random clusterings with equal cluster number
nsamples = 100
nclustersrange = np.linspace(2, nsamples/10, astype=int)
plt.figure(1)
plots = [
 names = [
 for scorefunc in score_funcs:
 print('Computing %s for values of nclusters and nsamples: %s' %
 (scorefunc.__name__, len(nclustersrange), nsamples))
 t0 = time()
 scores = uniform_labeling_score(scorefunc, nsamples, nclustersrange)

56 Clustering 887

```
scikitlearn user guide Release 0213
printdone in 03fs time t0
plotsappendplterrorbar
nclustersrange npmedianscores axis1 scoresstdaxis10
namesappendscorefuncname
plttitleClustering measures for 2 random uniform labelings n
with equal number of clusters
pltxlabelNumber of clusters Number of samples is fixed to d nsamples
pltylabelScore value
pltlegendplots names
pltylimbottom005 top105
Random labeling with varying nclusters against ground class labels
with fixed number of clusters
nsamples 1000
nclustersrange nplinspace2 100 10astypenpint
nclasses 10
pltfigure2
plots
names
forscorefunc inscorefuncs
printComputing sfordvalues of nclusters and nsamples d
scorefuncname lennclustersrange nsamples
t0 time
scores uniformlabelingsscoresscorefunc nsamples nclustersrange
fixednclasesnclases
printdone in 03fs time t0
plotsappendplterrorbar
nclustersrange scoresmeanaxis1 scoresstdaxis10
namesappendscorefuncname
plttitleClustering measures for random uniform labeling n
against reference assignment with dclases nclases
pltxlabelNumber of clusters Number of samples is fixed to d nsamples
pltylabelScore value
pltylimbottom005 top105
pltlegendplots names
pltshow
Total running time of the script 0 minutes 0835 seconds
Note Click here to download the full example code
5622 Feature agglomeration vs univariate selection
This example compares 2 dimensionality reduction strategies
• univariate feature selection with Anova
• feature agglomeration with Ward hierarchical clustering
888 Chapter 5 Examples
```

scikitlearn user guide Release 0213
Both methods are compared in a regression problem using a BayesianRidge as supervised estimator
Out

Memory Calling sklearnclusterhierarchicalwardtree
wardtreearray0451933 0675318

0275706 1085711
1600x1600 sparse matrix of type class numpyint64
with 7840 stored elements in COOrdinate format nclustersNone return
↪distanceFalse
wardtree 01s 00min

Memory Calling sklearnclusterhierarchicalwardtree
wardtreearray 0905206 0161245

0849835 1091621
1600x1600 sparse matrix of type class numpyint64
with 7840 stored elements in COOrdinate format nclustersNone return
↪distanceFalse
wardtree 01s 00min

Memory Calling sklearnclusterhierarchicalwardtree
wardtreearray 0905206 0675318

0849835 1085711
1600x1600 sparse matrix of type class numpyint64
with 7840 stored elements in COOrdinate format nclustersNone return
↪distanceFalse
wardtree 01s 00min

Memory Calling sklearnfeatureselectionunivariateselectionfregression
fregressionarray0451933 0275706

0675318 1085711
array 25267703 25026711
fregression 00s 00min

Memory Calling sklearnfeatureselectionunivariateselectionfregression
fregressionarray 0905206 0849835

56 Clustering 889

scikitlearn user guide Release 0213
0161245 1091621
array 27447268 112638768
fregression 00s 00min

Memory Calling sklearnfeatureselectionunivariateselectionfregression
fregressionarray 0905206 0849835

0675318 1085711
array27447268 25026711
fregression 00s 00min
Author Alexandre Gramfort alexandregamfortinriafr
License BSD (Binary Search Decision) 3 clause
printdoc
import shutil
import tempfile
import numpy as np
import matplotlib.pyplot as plt
from scipy import linalg ndimage
from joblib import Memory
from sklearnfeatureextractionimage import gridtograph
from sklearn import featureselection
from sklearncluster import FeatureAgglomeration
from sklearnlinearmodel import BayesianRidge
from sklearnpipeline import Pipeline
from sklearnmodelselection import GridSearchCV
from sklearnmodelselection import KFold

Generate data
nsamples 200
size 40 image size
roisize 15
snr 5
nprandomseed0
mask nponessize size dtype npbool
coef npzerossize size
coef0roisize 0roisize 1
coefroisize roisize 1
X nprandomrandnnsamples size 2
forxinX smooth data
x ndimagegaussianfilterxreshapesize size sigma10ravel
X Xmeanaxis0
X Xstdaxis0
y npdotX coefravel
noise nprandomrandnyshape0
890 Chapter 5 Examples


```
scikitlearn user guide Release 0213
noisecoef linalgnormy 2 npexpsnr 20 linalgnormnoise 2
y noisecoef noise add noise

Compute the coefs of a Bayesian Ridge with GridSearch
cv KFold2 crossvalidation generator for model selection
ridge BayesianRidge
cachedir tempfilemkdtemp
mem Memorylocationcachedir verbose1
Ward agglomeration followed by BayesianRidge
connectivity gridtographnxsize nysize
ward FeatureAgglomerationnclusters10 connectivityconnectivity
memorymem
clf Pipelineward ward ridge ridge
Select the optimal number of parcels with grid search
clf GridSearchCVclf wardnclusters 10 20 30 njobs1 cvcv
clffitX y set the best parameters
coef clfbestestimatorsteps11coef
coef clfbestestimatorsteps01inversetransformcoef
coefagglomeration coefreshapesize size
Anova univariate feature selection followed by BayesianRidge
fregression memcachefeatureselectionfregression caching function
anova featureselectionSelectPercentilefregression
clf Pipelineanova anova ridge ridge
Select the optimal percentage of features with grid search
clf GridSearchCVclf anovapercentile 5 10 20 cvcv
clffitX y set the best parameters
coef clfbestestimatorsteps11coef
coef clfbestestimatorsteps01inversetransformcoefreshape1 1
coefselection coefreshapesize size

Inverse the transformation to plot the results on an image
pltcloseall
pltfigurefigsize73 27
pltsubplot1 3 1
pltimshowcoef interpolationnearest cmappltcmRdBur
plttitleTrue weights
pltsubplot1 3 2
pltimshowcoefselection interpolationnearest cmappltcmRdBur
plttitleFeature Selection
pltsubplot1 3 3
pltimshowcoefagglomeration interpolationnearest cmappltcmRdBur
plttitleFeature Agglomeration
pltsubplotsadjust004 00 098 094 016 026
pltshow
Attempt to remove the temporary cachedir but dont worry if it fails
shutilrmtreecachedir ignoreerrorsTrue
Total running time of the script 0 minutes 0623 seconds
Note Click here to download the full example code
56 Clustering 891
```

scikitlearn user guide Release 0213

5623 Comparison of the KMeans and MiniBatchKMeans clustering algorithms

We want to compare the performance of the MiniBatchKMeans and KMeans the MiniBatchKMeans is faster but gives slightly different results see Mini Batch KMeans

We will cluster a set of data first with KMeans and then with MiniBatchKMeans and plot the results We will also plot the points that are labelled differently between the two algorithms

```
printdoc
import time
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import MiniBatchKMeans KMeans
from sklearn.metrics.pairwise import pairwise_distances_argmin
from sklearn.datasets.samples_generator import make_blobs

Generate sample data
np.random.seed(0)
batch_size = 45
centers = 1 1 1 1 1
n_clusters = len(centers)
X, labels = make_blobs(n_samples=3000, centers=centers, cluster_std=0.7
```

```
Compute clustering with Means
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=0)
t0 = time.time()
kmeans.fit(X)
tbatch = time.time() - t0
```

```
Compute clustering with MiniBatchKMeans
mbk = MiniBatchKMeans(n_clusters=3, batch_size=batch_size,
n_init=10, max_no_improvement=10, verbose=0)
t0 = time.time()
892 Chapter 5 Examples
```

scikitlearn user guide Release 0213
mbkfitX
tminibatch timetime t0

Plot result
fig pltfigurefigsize8 3
figsubplotsadjustleft002 right098 bottom005 top09
colors 4EACC5 FF9C34 4E9A06
We want to have the same colors for the same cluster from the
MiniBatchKMeans and the KMeans algorithm Lets pair the cluster centers per
closest one
kmeansclustercenters npsortkmeansclustercenters axis0
mbkmeansclustercenters npsortmbkclustercenters axis0
kmeanslabels pairwisedistancesargminX kmeansclustercenters
mbkmeanslabels pairwisedistancesargminX mbkmeansclustercenters
order pairwisedistancesargminkmeansclustercenters
mbkmeansclustercenters
KMeans
ax figaddsubplot1 3 1
fork colinziprangencusters colors
mymembers kmeanslabels k
clustercenter kmeansclustercentersk
axplotXmymembers 0 Xmymembers 1 w
markerfacecolorcol marker
axplotclustercenter0 clustercenter1 o markerfacecolorcol
markeredgecolorcol markersize6
axsettitleKMeans
axsetxticks
axsetyticks
plttext35 18 train time 2fsninertia f
tbatch kmeansinertia
MiniBatchKMeans
ax figaddsubplot1 3 2
fork colinziprangencusters colors
mymembers mbkmeanslabels orderk
clustercenter mbkmeansclustercentersorderk
axplotXmymembers 0 Xmymembers 1 w
markerfacecolorcol marker
axplotclustercenter0 clustercenter1 o markerfacecolorcol
markeredgecolorcol markersize6
axsettitleMiniBatchKMeans
axsetxticks
axsetyticks
plttext35 18 train time 2fsninertia f
tminibatch mbkinertia
Initialise the different array to all False
different mbkmeanslabels 4
ax figaddsubplot1 3 3
forkinrangencusters
different kmeanslabels k mbkmeanslabels orderk
identic nplogicalnotdifferent
56 Clustering 893

scikitlearn user guide Release 0213

axplotXidentic 0 Xidentic 1 w

markerfacecolorbbbbbb marker

axplotXdifferent 0 Xdifferent 1 w

markerfacecolorm marker

axsettitleDifference

axsetxticks

axsetyticks

pltshow

Total running time of the script 0 minutes 0127 seconds

Note Click here to download the full example code

5624 A demo of KMeans clustering on the handwritten digits data

In this example we compare the various initialization strategies for Kmeans in terms of runtime and quality of the

results

As the ground truth is known here we also apply different cluster quality metrics to judge the goodness of fit of the

cluster labels to the ground truth

Cluster quality metrics evaluated see Clustering performance evaluation for definitions and discussions of the met

rics

Shorthand full name

homo homogeneity score

compl completeness score

vmeas V measure

ARI adjusted Rand index

AMI adjusted mutual information

silhouette silhouette coefficient

894 Chapter 5 Examples

scikitlearn user guide Release 0213

Out

ndigits 10 nsamples 1797 nfeatures 64

init time inertia homo compl vmeas ARI AMI silhouette
kmeans 037s 69432 0602 0650 0625 0465 0621 0146
random 027s 69694 0669 0710 0689 0553 0686 0147
PCAbased 003s 70804 0671 0698 0684 0561 0681 0118

printdoc
from time import time
import numpy as np
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearncluster import KMeans
from sklearndatasets import loaddigits
from sklearndecomposition import PCA (Principal Component Analysis)
from sklearnpreprocessing import scale
56 Clustering 895

```
scikitlearn user guide Release 0213
nprandomseed42
digits loaddigits
data scaledigitsdata
nsamples nfeatures datashape
ndigits lennpuniquedigitstarget
labels digitstarget
samplesize 300
printndigits dtnsamples dtfeatures d
ndigits nsamples nfeatures
print82
printinittttimetinertiathomotcompltvmeastARitAMitsilhouette
defbenchkmeansestimator name data
t0 time
estimatorfitdata
print9st2fst3ft3ft3ft3ft3f
name time t0 estimatorinertia
metricsshomogeneityscorelabels estimatorlabels
metricscompletenessscorelabels estimatorlabels
metricssvmeasurescorelabels estimatorlabels
metricsadjustedrandscorelabels estimatorlabels
metricsadjustedmutualinfoscorelabels estimatorlabels
averagemethodarithmetic
metricssilhouettescoredata estimatorlabels
metricseuclidean
samplesizesamplesize
benchkmeansKMeansinitkmeans nclustersndigits ninit10
namekmeans datadata
benchkmeansKMeansinitrandom nclustersndigits ninit10
namerandom datadata
in this case the seeding of the centers is deterministic hence we run the
kmeans algorithm only once with ninit1
pca PCAncomponentsndigitsfitdata
benchkmeansKMeansinitpcacomponents nclustersndigits ninit1
namePCAbased
datadata
print82

Visualize the results on PCAreduced data
reduceddata PCAncomponents2fittransformdata
kmeans KMeansinitkmeans nclustersndigits ninit10
kmeansfitreduceddata
Step size of the mesh Decrease to increase the quality of the VQ
h 02 point in the mesh xmin xmaxxmin ymax
896 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Plot the decision boundary For that we will assign a color to each

xmin xmax reduceddata 0min 1 reduceddata 0max 1

ymin ymax reduceddata 1min 1 reduceddata 1max 1

xx yy npmeshgridnparangexmin xmax h nparangeymin ymax h

Obtain labels for each point in mesh Use last trained model

Z kmeanspredictnpcxxravel yy.ravel

Put the result into a color plot

Z Z.reshapexxshape

pltfigure1

pltclf

pltimshowZ interpolationnearest

extentxxmin xmax yymin ymax

cmappltcmPaired

aspectauto originlower

pltplotreduceddata 0 reduceddata 1 k markersize2

Plot the centroids as a white X

centroids kmeansclustercenters

pltscattercentroids 0 centroids 1

markerx s169 linewidths3

colorw zorder10

plttitleKmeans clustering on the digits dataset PCAreduced data n

Centroids are marked with white cross

pltxlimxmin xmax

pltylimymin ymax

pltxticks

pltyticks

pltshow

Total running time of the script 0 minutes 1230 seconds

Note Click here to download the full example code

5625 Comparing different hierarchical linkage methods on toy datasets

This example shows characteristics of different linkage methods for hierarchical clustering on datasets that are “inter

esting” but still in 2D

The main observations to make are

- single linkage is fast and can perform well on nonglobular data but it performs poorly in the presence of noise
- average and complete linkage perform well on cleanly separated globular clusters but have mixed results other wise
- Ward is the most effective method for noisy data

While these examples give some intuition about the algorithms this intuition might not apply to very high dimensional data

printdoc

import time

56 Clustering 897

```
scikitlearn user guide Release 0213
import warnings
import numpy as np
import matplotlib.pyplot as plt
from sklearn import cluster datasets
from sklearn.preprocessing import StandardScaler
from itertools import cycle islice
nprandomseed0
Generate datasets We choose the size big enough to see the scalability of the algorithms but not too big to avoid too
long running times
nsamples 1500
noisycircles datasetsmakecirclesnsamplesnsamples factor5
noise05
noisymoons datasetsmakemoonsnsamplesnsamples noise05
blobs datasetsmakeblobsnsamplesnsamples randomstate8
nostructure nprandomrandnsamples 2 None
Anisotropically distributed data
randomstate 170
X y datasetsmakeblobsnsamplesnsamples randomstaterandomstate
transformation 06 06 04 08
Xaniso npdotX transformation
aniso Xaniso y
blobs with varied variances
varied datasetsmakeblobsnsamplesnsamples
clusterstd10 25 05
randomstaterandomstate
Run the clustering and plot
Set up cluster parameters
pltfigurefigsize9 13 2 145
pltsubplotsadjustleft02 right98 bottom001 top96 wspace05
hspace01
plotnum 1
defaultbase nneighbors 10
nclusters 3
datasets
noisycircles nclusters 2
noisymoons nclusters 2
varied nneighbors 2
aniso nneighbors 2
blobs
nostructure
foridataset dataset algoparams inenumerateddatasets
update parameters with datasetspecific values
params defaultbasecopy
paramsupdatealgoparams
898 Chapter 5 Examples
```


scikitlearn user guide Release 0213
X y dataset
normalize dataset for easier parameter selection
X StandardScalerfittransformX

Create cluster objects

ward clusterAgglomerativeClustering
nclustersparamsnclusters linkageward
complete clusterAgglomerativeClustering
nclustersparamsnclusters linkagecomplete
average clusterAgglomerativeClustering
nclustersparamsnclusters linkageaverage
single clusterAgglomerativeClustering
nclustersparamsnclusters linkagesingle
clusteringalgorithms
Single Linkage single
Average Linkage average
Complete Linkage complete
Ward Linkage ward

forname algorithm inclusteringalgorithms
t0 time
catch warnings related to kneighborsgraph
withwarningscatchwarnings
warningsfilterwarnings
ignore
message the number of connected components of the
connectivity matrix is 0912
1 Completing it to avoid stopping the tree early
categoryUserWarning
algorithmfitX
t1 time
ifhasattralgorithm labels
ypred algorithmlabelsastypenpint
else
ypred algorithmpredictX
pltsubplotlendatasets lenclusteringalgorithms plotnum
ifidataset 0
plttitle name size18
colors nparraylistislice cycle377eb8 ff7f00 4daf4a
f781bf a65628 984ea3
999999 e41a1c dede00
intmaxypred 1
pltscatterX 0 X 1 s10 colorcolorscopyred
pltxlim25 25
pltylim25 25
pltxticks
pltyticks
plttext99 01 2fs t1 t0lstrip0
56 Clustering 899

scikitlearn user guide Release 0213
transformpltgcatransAxes size15
horizontalalignmentright
plotnum 1
pltshow
Total running time of the script 0 minutes 1577 seconds
Note [Click here to download the full example code](#)
900 Chapter 5 Examples

5626 Selecting the number of clusters with silhouette analysis on KMeans clustering

Silhouette analysis can be used to study the separation distance between the resulting clusters. The silhouette plot displays a measure of how close each point in one cluster is to points in the neighboring clusters and thus provides a way to assess parameters like number of clusters visually. This measure has a range of 1. 1. Silhouette coefficients as these values are referred to as near 1 indicate that the sample is far away from the neighboring clusters. A value of 0 indicates that the sample is on or very close to the decision boundary between two neighboring clusters and negative values indicate that those samples might have been assigned to the wrong cluster. In this example the silhouette analysis is used to choose an optimal value for nclusters. The silhouette plot shows that thenclusters value of 3, 5 and 6 are a bad pick for the given data due to the presence of clusters with below average silhouette scores and also due to wide fluctuations in the size of the silhouette plots. Silhouette analysis is more ambivalent in deciding between 2 and 4. Also from the thickness of the silhouette plot the cluster size can be visualized. The silhouette plot for cluster 0 when nclusters is equal to 2 is bigger in size owing to the grouping of the 3 sub clusters into one big cluster. However when thenclusters is equal to 4 all the plots are more or less of similar thickness and hence are of similar sizes as can be also verified from the labelled scatter plot on the right.

-
-

scikitlearn user guide Release 0213

-
-
-

Out

For nclusters 2 The average silhouettescore is 07049787496083262
For nclusters 3 The average silhouettescore is 05882004012129721
For nclusters 4 The average silhouettescore is 06505186632729437
For nclusters 5 The average silhouettescore is 056376469026194
For nclusters 6 The average silhouettescore is 04504666294372765
902 Chapter 5 Examples

```
scikitlearn user guide Release 0213
from sklearn.datasets import makeblobs
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_samples, silhouette_score
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import numpy as np
printdoc
Generating the sample data from makeblobs
This particular setting has one distinct cluster and 3 clusters placed close
together
X, y = makeblobs(n_samples=500,
n_features=2,
centers=4,
cluster_std=1,
center_box=(100, 100),
shuffle=True,
random_state=1) # For reproducibility
range_clusters = [2, 3, 4, 5, 6]
for n_clusters in range_clusters:
    Create a subplot with 1 row and 2 columns
    fig, ax = plt.subplots(1, 2)
    fig.set_size_inches(18, 7)
    The 1st subplot is the silhouette plot
    The silhouette coefficient can range from -1 to 1 but in this example all
    lie within [0, 1]
    ax[0].set_xlim(0, 1)
    The n_clusters=10 is for inserting blank space between silhouette
    plots of individual clusters to demarcate them clearly
    ax[1].set_ylim(0, len(X) - n_clusters + 1)
    Initialize the clusterer with n_clusters value and a random generator
    seed of 10 for reproducibility
    clusterer = KMeans(n_clusters=n_clusters, random_state=10)
    cluster_labels = clusterer.fit_predict(X)
    The silhouette_score gives the average value for all the samples
    This gives a perspective into the density and separation of the formed
    clusters
    silhouette_avg = silhouette_score(X, cluster_labels)
    print('For n_clusters =', n_clusters,
    The average silhouette_score is:', silhouette_avg)
    Compute the silhouette scores for each sample
    samples_silhouette_values = silhouette_samples(X, cluster_labels)
    y_lower = 10
    for i in range_clusters:
        Aggregate the silhouette scores for samples belonging to
        cluster i and sort them
56 Clustering 903
```

```
scikitlearn user guide Release 0213
ithclustersilhouettevalues
samplesilhouettevaluesclusterlabels i
ithclustersilhouettevaluessort
sizeclusteri ithclustersilhouettevaluesshape0
yupper ylower sizeclusteri
color cmnipyspectralfloati nclusters
ax1fillbetweenxnparangeylower yupper
0 ithclustersilhouettevalues
facecolorcolor edgecolorcolor alpha07
Label the silhouette plots with their cluster numbers at the middle
ax1text005 ylower 05 sizeclusteri stri
Compute the new ylower for next plot
ylower yupper 10 10 for the 0 samples
ax1settitleThe silhouette plot for the various clusters
ax1setxlabelThe silhouette coefficient values
ax1setylabelCluster label
The vertical line for average silhouette score of all the values
ax1axvlinexsilhouetteavg colored linestyle
ax1setyticks Clear the yaxis labels ticks
ax1setxticks01 0 02 04 06 08 1
2nd Plot showing the actual clusters formed
colors cmnipyspectralclusterlabelsastypefloat nclusters
ax2scatterX 0 X 1 marker s30 lw0 alpha07
ccolors edgecolork
Labeling the clusters
centers clustererclustercenters
Draw white circles at cluster centers
ax2scattercenters 0 centers 1 markero
cwhite alpha1 s200 edgecolork
fori cinenumeratecenters
ax2scatterc0 c1 marker d i alpha1
s50 edgecolork
ax2settitleThe visualization of the clustered data
ax2setxlabelFeature space for the 1st feature
ax2setylabelFeature space for the 2nd feature
pltstitleSilhouette analysis for KMeans clustering on sample data
with nclusters d nclusters
fontsize14 fontweightbold
pltshow
Total running time of the script 0 minutes 0629 seconds
904 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Note [Click here to download the full example code](#)

5627 Comparing different clustering algorithms on toy datasets

This example shows characteristics of different clustering algorithms on datasets that are “interesting” but still in 2D

With the exception of the last dataset the parameters of each of these datasetalgorithm pairs has been tuned to produce good clustering results Some algorithms are more sensitive to parameter values than others

The last dataset is an example of a ‘null’ situation for clustering the data is homogeneous and there is no good clustering For this example the null dataset uses the same parameters as the dataset in the row above it which

represents a mismatch in the parameter values and the data structure

While these examples give some intuition about the algorithms this intuition might not apply to very high dimensional data

```
printdoc
import time
import warnings
import numpy as np
import matplotlib.pyplot as plt
from sklearn import cluster datasets mixture
from sklearnneighbors import kneighborsgraph
from sklearnpreprocessing import StandardScaler
from itertools import cycle islice
nprandomseed0
56 Clustering 905
```

Generate datasets We choose the size big enough to see the scalability of the algorithms but not too big to avoid too long running times

```
nsamples 1500
noisycircles datasetsmakecirclesnsamplesnsamples factor5
noise05
noisymoons datasetsmakemoonsnsamplesnsamples noise05
blobs datasetsmakeblobsnsamplesnsamples randomstate8
nostructure nprandomrandnsamples 2 None
Anisotropically distributed data
randomstate 170
X y datasetsmakeblobsnsamplesnsamples randomstaterandomstate
transformation 06 06 04 08
Xaniso npdotX transformation
aniso Xaniso y
blobs with varied variances
varied datasetsmakeblobsnsamplesnsamples
clusterstd10 25 05
randomstaterandomstate
```

Set up cluster parameters

```
pltfigurefigsize9 2 3 125
pltsubplotsadjustleft02 right98 bottom001 top96 wspace05
hspace01
plotnum 1
defaultbase quantile 3
eps 3
damping 9
preference 200
nneighbors 10
nclusters 3
minsamples 20
xi 005
minclustersize 01
datasets
noisycircles damping 77 preference 240
quantile 2 nclusters 2
minsamples 20 xi 025
noisymoons damping 75 preference 220 nclusters 2
varied eps 18 nneighbors 2
minsamples 5 xi 0035 minclustersize 2
aniso eps 15 nneighbors 2
minsamples 20 xi 01 minclustersize 2
blobs
nostructure
foridataset dataset algoparams inenumerateddatasets
update parameters with datasetspecific values
params defaultbasecopy
906 Chapter 5 Examples
```


scikitlearn user guide Release 0213
paramsupdatealgoparams
X y dataset
 normalize dataset for easier parameter selection
X StandardScalerfittransformX
 estimate bandwidth for mean shift
bandwidth clusterestimatebandwidthX quantileparamsquantile
 connectivity matrix for structured Ward
connectivity kneighborsgraph
X nneighborsparamsnneighbors includeselfFalse
 make connectivity symmetric
connectivity 05 connectivity connectivityT

Create cluster objects

ms clusterMeanShiftbandwidthbandwidth binseedingTrue
twomeans clusterMiniBatchKMeansnclustersparamsnclusters
ward clusterAgglomerativeClustering
nclustersparamsnclusters linkageward
connectivityconnectivity
spectral clusterSpectralClustering
nclustersparamsnclusters eigensolverarpack
affinitynearestneighbors
dbscan clusterDBSCANepsparamseps
optics clusterOPTICSminsamplesparamsminsamples
xiparamsxi
minclustersizeparamsminclustersize
affinitypropagation clusterAffinityPropagation
dampingparamsdamping preferenceparamspreference
averagelinkage clusterAgglomerativeClustering
linkageaverage affinitycityblock
nclustersparamsnclusters connectivityconnectivity
birch clusterBirchnclustersparamsnclusters
gmm mixtureGaussianMixture
ncomponentsparamsnclusters covariancetypefull
clusteringalgorithms
MiniBatchKMeans twomeans
AffinityPropagation affinitypropagation
MeanShift ms
SpectralClustering spectral
Ward ward
AgglomerativeClustering averagelinkage
DBSCAN dbscan
OPTICS optics
Birch birch
GaussianMixture gmm

forname algorithm inclusteringalgorithms
t0 timetime
 catch warnings related to kneighborsgraph
withwarningscatchwarnings
56 Clustering 907

scikitlearn user guide Release 0213

warningsfilterwarnings

ignore

message the number of connected components of the connectivity matrix is 0912

1 Completing it to avoid stopping the tree early

categoryUserWarning

warningsfilterwarnings

ignore

messageGraph is not fully connected spectral embedding may not work as expected

categoryUserWarning

algorithmfitX

t1 timetime

ifhasattralgorithm labels

ypred algorithmlabelsastypenpint

else

ypred algorithmpredictX

pltsubplotlendatasets lenclusteringalgorithms plotnum

ifidataset 0

plttitle name size18

colors nparraylistislice cycle377eb8 ff7f00 4daf4a f781bf a65628 984ea3 999999 e41a1c dede00

intmaxypred 1

add black color for outliers if any

colors npappendcolors 000000

pltscatterX 0 X 1 s10 colorcolorsyypred

pltxlim25 25

pltylim25 25

pltxticks

pltyticks

plttext99 01 2fs t1 t0lstrip0

transformpltgcatransAxes size15

horizontalalignmentright

plotnum 1

pltshow

Total running time of the script 0 minutes 39530 seconds

57 Pipelines and composite estimators

Examples of how to compose transformers and pipelines from other estimators See the User Guide

Note Click here to download the full example code

908 Chapter 5 Examples

571 Concatenating multiple feature extraction methods

In many realworld examples there are many ways to extract features from a dataset Often it is beneficial to combine several methods to obtain good performance This example shows how to use FeatureUnion to combine features obtained by PCA (Principal Component Analysis) and univariate selection

Combining features using this transformer has the benefit that it allows cross validation and grid searches over the whole process

The combination used in this example is not particularly helpful on this dataset and is only used to illustrate the usage of FeatureUnion

Out

Combined space has 3 features

Fitting 5 folds for each of 18 candidates totalling 90 fits

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01 score0

↪933 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01 score0

↪933 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01 score0

↪867 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01 score0

↪933 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC01 score1

↪000 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1 score0

↪900 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1 score1

↪000 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1 score0

↪867 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1 score0

↪933 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC1 score1

↪000 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10 score0

↪933 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10 score1

↪000 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10 score0

↪900 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10 score0

↪933 total 00s

CV (Cross-Validation) featurespcancomponents1 featuresunivselectk1 svmC10

[illegible]

[illegible]

[illegible]

scikitlearn user guide Release 0213

CV (Cross-Validation) featurespcancomponents3 featuresunivselectk1 svmC10
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk1 svmC10 score0
↪967 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk1 svmC10
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk1 svmC10 score1
↪000 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01 score0
↪967 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01 score1
↪000 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01 score0
↪933 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01 score0
↪967 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC01 score1
↪000 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC1
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC1 score0
↪967 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC1
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC1 score1
↪000 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC1
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC1 score0
↪967 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC1
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC1 score1
↪000 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10 score1
↪000 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10 score1
↪000 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10 score0
↪900 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10 score0
↪967 total 00s
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10
CV (Cross-Validation) featurespcancomponents3 featuresunivselectk2 svmC10 score1
↪000 total 00s
Pipelinstepsfeatures
FeatureUniontransformerlistpca PCAncomponents3
univselect
SelectKBestk1
svm SVCC10 kernellinear
57 Pipelines and composite estimators 913

scikitlearn user guide Release 0213
Author Andreas Mueller amuelleraisunibonnde

License BSD (Binary Search Decision) 3 clause
from sklearnpipeline import Pipeline FeatureUnion
from sklearnmodelselection import GridSearchCV
from sklearnsvm import SVC (Support Vector Classification)
from sklearndatasets import loadiris
from sklearndecomposition import PCA (Principal Component Analysis)
from sklearnfeatureselection import SelectKBest
iris loadiris
X y irisdata iristarget
This dataset is way too highdimensional Better do PCA (Principal Component Analysis)
pca PCAncomponents2
Maybe some original features where good too
selection SelectKBestk1
Build estimator from PCA (Principal Component Analysis) and Univariate selection
combinedfeatures FeatureUnionpca pca univselect selection
Use combined features to transform dataset
Xfeatures combinedfeaturesfitX ytransformX
printCombined space has Xfeaturesshape1 features
svm SVCkernellinear
Do grid search over k ncomponents and C
pipeline Pipelinefeatures combinedfeatures svm svm
paramgrid dictfeaturespcancomponents1 2 3
featuresunivselectk1 2
svmC01 1 10
gridsearch GridSearchCVpipeline paramgridparamgrid cv5 verbose10
gridsearchfitX y
printgridsearchbestestimator
Total running time of the script 0 minutes 0865 seconds
Note Click here to download the full example code
572 Pipelining chaining a PCA (Principal Component Analysis) and a logistic regression
The PCA (Principal Component Analysis) does an unsupervised dimensionality reduction while the logistic regression does the prediction
We use a GridSearchCV to set the dimensionality of the PCA (Principal Component Analysis)
914 Chapter 5 Examples

scikitlearn user guide Release 0213
Out
Best parameter CV (Cross-Validation) score0917
logisticalpha 001 pcancomponents 64
printdoc
Code source Gaël Varoquaux
Modified for documentation by Jaques Grobler
License BSD (Binary Search Decision) 3 clause
57 Pipelines and composite estimators 915

```
scikitlearn user guide Release 0213
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import datasets
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.linear_model import SGDClassifier
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
Define a pipeline to search for the best combination of PCA (Principal Component Analysis) truncation
and classifier regularization
logistic SGDClassifier(loss=log_loss, penalty=l2, early_stopping=True
max_iter=10000, tol=1e-5, random_state=0)
pca = PCA (Principal Component Analysis)
pipe = Pipeline(steps=[pca, logistic])
digits = datasets.load_digits()
X_digits, y_digits = digits.data, digits.target
Parameters of pipelines can be set using “ ” separated parameter names
param_grid = {
    'pca__n_components': [5, 20, 30, 40, 50, 64],
    'logistic__alpha': np.logspace(-4, 4, 5)
}
search = GridSearchCV(pipe, param_grid, iid=False, cv=5)
search.fit(X_digits, y_digits)
print('Best parameter CV (Cross-Validation) score: %f' % search.best_score_)
print('Search best parameters: %s' % search.best_params_)
Plot the PCA (Principal Component Analysis) spectrum
pca.fit(X_digits)
fig, (ax0, ax1) = plt.subplots(nrows=2, sharex=True, figsize=(6, 6))
ax0.plot(pca.explained_variance_ratio_, linewidth=2)
ax0.set_ylabel('PCA explained variance')
ax0.axvline(search.best_estimator_.named_steps['pca'].n_components)
linestyle, label, n_components = chosen
ax0.legend(prop=dict(size=12))
For each number of components find the best classifier results
results = pd.DataFrame(search.cv_results_)
components_col = param['pca__n_components']
best_clfs = results.groupby(components_col).apply(
    lambda g: g.nlargest(1, 'mean_test_score')
)
best_clf = plot_x(components_col, y_mean_test_score, y_err_std_test_score)
legend = False
ax1.set_ylabel('Classification accuracy')
ax1.set_xlabel('Number of components')
plt.tight_layout()
plt.show()
Total running time of the script: 0 minutes 20.748 seconds
916 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Note Click here to download the full example code

573 Column Transformer with Mixed Types

This example illustrates how to apply different preprocessing and feature extraction pipelines to different subsets of features using `sklearn.compose.ColumnTransformer`. This is particularly handy for the case of datasets that contain heterogeneous data types since we may want to scale the numeric features and onehot encode the categorical ones.

In this example the numeric data is standard scaled after mean imputation while the categorical data is onehot encoded after imputing missing values with a new category `missing`.

Finally the preprocessing pipeline is integrated in a full prediction pipeline using `sklearn.pipeline`.

Pipeline together with a simple classification model.

Author Pedro Morales partmorales@gmail.com

License BSD (Binary Search Decision) 3 clause

import pandas as pd

import numpy as np

from sklearn.compose import ColumnTransformer

from sklearn.pipeline import Pipeline

from sklearn.impute import SimpleImputer

from sklearn.preprocessing import StandardScaler, OneHotEncoder

from sklearn.linear_model import LogisticRegression

from sklearn.model_selection import train_test_split, GridSearchCV

np.random.seed(0)

Read data from Titanic dataset

titanic_url = <https://raw.githubusercontent.com/amueller/scipy2017-sklearn/091d371/notebooks/dataset/titanic3.csv>

data = pd.read_csv(titanic_url)

We will train our classifier with the following features:

Numeric Features

age float

fare float

Categorical Features

embarked categories encoded as strings C S Q

sex categories encoded as strings female male

pclass ordinal integers 1 2 3

We create the preprocessing pipelines for both numeric and categorical data

numeric_features = age fare

numeric_transformer = Pipeline(steps=

imputer=SimpleImputer(strategy='median')

scaler=StandardScaler())

categorical_features = embarked sex pclass

categorical_transformer = Pipeline(steps=

imputer=SimpleImputer(strategy='constant', fill_value='missing')

onehot=OneHotEncoder(handle_unknown='ignore')

57 Pipelines and composite estimators 917

```
scikitlearn user guide Release 0213
preprocessor ColumnTransformer
transformers
num numerictransformer numericfeatures
cat categoricaltransformer categoricalfeatures
Append classifier to preprocessing pipeline
Now we have a full prediction pipeline
clf Pipeline(steps=[preprocessor, preprocessor,
classifier LogisticRegression(solver='lbfgs')])
X datadropsurvived axis1
y datasurvived
Xtrain Xtest ytrain ytest train_test_split(X, y, test_size=0.2)
clf.fit(Xtrain, ytrain)
print(model.score(Xtest, ytest))
Out
model score 0.790
Using the prediction pipeline in a grid search
Grid search can also be performed on the different preprocessing steps defined in the
ColumnTransformer object together with the classifier's hyperparameters as part of the Pipeline
We will search for both the imputer strategy of the numeric preprocessing and the regularization parameter
of the logistic regression using sklearn.model_selection.GridSearchCV
param_grid = {
    'preprocessor__num__imputer__strategy': ['mean', 'median'],
    'classifier__C': [0.1, 1.0, 10.0, 100.0]
}
gridsearch = GridSearchCV(clf, param_grid, cv=10, iid=False)
gridsearch.fit(Xtrain, ytrain)
print('best logistic regression from grid search %f' % gridsearch.score(Xtest, ytest))
Out
best logistic regression from grid search 0.798
Total running time of the script: 0 minutes 21.03 seconds
Note: Click here to download the full example code
574 Selecting dimensionality reduction with Pipeline and GridSearchCV
This example constructs a pipeline that does dimensionality reduction followed by prediction with a support vector
classifier. It demonstrates the use of GridSearchCV and Pipeline to optimize over different classes of estimators.
918 Chapter 5 Examples
```

scikitlearn user guide Release 0213

in a single CV (Cross-Validation) run – unsupervised PCA (Principal Component Analysis) andNMF dimensionality reductions are compared to univariate during the grid search

Additionally Pipeline can be instantiated with the memory argument to memoize the transformers within the pipeline avoiding to fit again the same transformers over and over

Note that the use of memory to enable caching becomes interesting when the fitting of a transformer is costly

Illustration of Pipeline andGridSearchCV

This section illustrates the use of a Pipeline withGridSearchCV

Authors Robert McGibbon Joel Nothman Guillaume Lemaitre

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_digits
from sklearn.model_selection import GridSearchCV
from sklearn.pipeline import Pipeline
from sklearn.svm import LinearSVC
from sklearn.decomposition import PCA (Principal Component Analysis) NMF (Non-Negative Matrix Factorization)
from sklearn.feature_selection import SelectKBest chi2
printdoc
pipe Pipeline
    the reducedim stage is populated by the paramgrid
reducedim passthrough
classify LinearSVCdualFalse maxiter10000

NFEATURESOPTIONS 2 4 8
COPTIONS 1 10 100 1000
paramgrid

reducedim PCAiteratedpower7 NMF (Non-Negative Matrix Factorization)
reducedimncomponents NFEATURESOPTIONS
classifyC COPTIONS

reducedim SelectKBestchi2
reducedimk NFEATURESOPTIONS
classifyC COPTIONS

reducerlabels PCA (Principal Component Analysis) NMF (Non-Negative Matrix Factorization) KBestchi2
grid GridSearchCVpipe cv5 njobs1 paramgridparamgrid iidFalse
digits load_digits
gridfitdigitsdata digitstarget
meanscores nparraygridcvresultsmeantestscore
    scores are in the order of paramgrid iteration which is alphabetical
meanscores meanscoresreshapelenCOPTIONS 1 lenNFEATURESOPTIONS
    select score for best C
meanscores meanscoresmaxaxis0
baroffsets nparangelenNFEATURESOPTIONS
57 Pipelines and composite estimators 919
```

```
scikitlearn user guide Release 0213
lenreducerlabels 1 5
pltfigure
COLORS bgrcmyk
fori label reducerscores inenumeratezipreducerlabels meanscores
pltbarbaroffsets i reducerscores labellabel colorCOLORSi
plttitleComparing feature reduction techniques
pltxlabelReduced number of features
pltxticksbaroffsets lenreducerlabels 2 NFEATURESOPTIONS
pltylabelDigit classification accuracy
pltylim0 1
pltlegendlocupper left
pltshow
Caching transformers within a Pipeline
It is sometimes worthwhile storing the state of a specific transformer since it could be used again Using a
pipeline inGridSearchCV triggers such situations Therefore we use the argument memory to enable
caching
920 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Warning Note that this example is however only an illustration since for this specific case fitting PCA (Principal Component Analysis) is not necessarily slower than loading the cache Hence use the memory constructor parameter when the fitting of a transformer is costly

```
from tempfile import mkdtemp
from shutil import rmtree
from joblib import Memory

# Create a temporary folder to store the transformers of the pipeline
cachedir = mkdtemp()
memory = Memory(location=cachedir, verbose=10)
cachedpipe = Pipeline(reducedim=PCA(Principal Component Analysis))
classify = LinearSVC(dual=False, max_iter=10000)
memory.memory

# This time a cached pipeline will be used within the grid search
grid = GridSearchCV(cachedpipe, cv=5, n_jobs=1, param_grid=param_grid, iid=False)
digits = load_digits()
grid.fit(digits.data, digits.target)

# Delete the temporary cache before exiting
rmtree(cachedir)
Out
```

Memory Calling sklearn.pipeline.fit_transformone
fit_transformonePCAiteratedpower7 ncomponents2 array0 0

0 0 array0 8 None messageclsnamePipeline
↳messageNone
fit_transformone 00s 00min

Memory Calling sklearn.pipeline.fit_transformone
fit_transformonePCAiteratedpower7 ncomponents2 array0 0

0 0 array0 8 None messageclsnamePipeline
↳messageNone
fit_transformone 00s 00min

Memory Calling sklearn.pipeline.fit_transformone
fit_transformonePCAiteratedpower7 ncomponents2 array0 0

0 0 array0 8 None messageclsnamePipeline
↳messageNone
fit_transformone 00s 00min

Memory Calling sklearn.pipeline.fit_transformone
fit_transformonePCAiteratedpower7 ncomponents2 array0 0

0 0 array0 8 None messageclsnamePipeline
↳messageNone
fit_transformone 00s 00min

Memory Calling sklearn.pipeline.fit_transformone
57 Pipelines and composite estimators 921

scikitlearn user guide Release 0213
fittransformonePCAiteratedpower7 ncomponents2 array0 0

0 0 array0 9 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents4 array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents4 array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents4 array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents4 array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents4 array0 0

0 0 array0 9 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents8 array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 01s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents8 array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents8 array0 0

scikitlearn user guide Release 0213
0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents8 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents8 array0 0

0 0 array0 9 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents2 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 01s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents2 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents2 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 01s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents2 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents2 array0 0

0 0 array0 9 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents4 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
57 Pipelines and composite estimators 923

scikitlearn user guide Release 0213
fittransformone 01s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents4 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents4 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents4 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents4 array0 0

0 0 array0 9 None messageclsnamePipeline
↩→messageNone
fittransformone 01s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents8 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 01s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents8 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 01s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents8 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 02s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents8 array0 0

0 0 array0 8 None messageclsnamePipeline
↩→messageNone
fittransformone 01s 00min

scikitlearn user guide Release 0213

Memory Calling sklearnpipelinefittransformone
fittransformoneNMFncomponents8 array0 0

0 0 array0 9 None messageclsnamePipeline
↪messageNone

fittransformone 01s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone591acae8e60c9632aa990e4fa0962a67
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone323909961fcc2a4950defb581f08007f
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone3654e1d61587aee4f9980c99541d55d3
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone835820e75be3172b4e2e257028147bb2
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone90604ce8bb756e3ff1acd4c1ce065b1a
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone3c66df347f488dfe044ecf991ca2498b
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonebf4bec5fe5245bf3c0d271e31bc2342f
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonece0f911df93a8e38932c48f983e7fde6
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone285efba3f9e5b1a35d6825e37a718019
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone74083d6e1ed743fa1e756a41ba467e6c
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoneb72f745822a406f2191b6ebfd8ca9df9
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone997e4ce30ba5143e90330db664bf61c5
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone3140177f6ccbe72992772342347a166f
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonef843896ad42a20c4a50c0fd29512111a
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoned91a2d083b8c7311c87edd6592fcd446
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone50eb0f4152cd0f7dce3b471add31d07d
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone3b28284b39c934cb519f2a31e03a951b
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoneaded0eb932b9c692c801c6c56dc2bda2
57 Pipelines and composite estimators 925

scikitlearn user guide Release 0213

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformonecdf3d1530a8a2388ba80a16c66413409

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoneff5666a0cb943711a4e4dfd5b5cd8587

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone536a5fbb5a6c8d69b4fb426f0b51d9eb

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone26cd3a8242acb986f98cf38291ef5baa

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone8b6094f421dab9fc88fb54c2f32e16b4

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoned0e8598c62dcb5f059a5c2b541c23b09

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone34affb50a3d55c5302e1c7f97aec9679

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoneb3635522e8f89bbe9e56ab7e7f4693fc

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone7389ad9bae0ad1181b7da9e4dd39449d

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoneab59173e00ab4a7bdf7740e5c6d50123

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone42894538fd0ab2b4235e55ef53b9cd59

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone580596a10617556e39d5e069c4de096a

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone591acae8e60c9632aa990e4fa0962a67

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone323909961fcc2a4950defb581f08007f

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone3654e1d61587aee4f9980c99541d55d3

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone835820e75be3172b4e2e257028147bb2

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone90604ce8bb756e3ff1acd4c1ce065b1a

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone3c66df347f488dfe044ecf991ca2498b

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformonebf4bec5fe5245bf3c0d271e31bc2342f

fittransformone cache loaded 00s 00min

926 Chapter 5 Examples

scikitlearn user guide Release 0213

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonece0f911df93a8e38932c48f983e7fde6
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone285efba3f9e5b1a35d6825e37a718019
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone74083d6e1ed743fa1e756a41ba467e6c
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoneb72f745822a406f2191b6ebfd8ca9df9
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone997e4ce30ba5143e90330db664bf61c5
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone3140177f6ccbe72992772342347a166f
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonef843896ad42a20c4a50c0fd29512111a
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoned91a2d083b8c7311c87edd6592fcd446
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone50eb0f4152cd0f7dce3b471add31d07d
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone3b28284b39c934cb519f2a31e03a951b
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoneaded0eb932b9c692c801c6c56dc2bda2
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonecdf3d1530a8a2388ba80a16c66413409
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoneff5666a0cb943711a4e4dfd5b5cd8587
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone536a5fbb5a6c8d69b4fb426f0b51d9eb
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone26cd3a8242acb986f98cf38291ef5baa
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone8b6094f421dab9fc88fb54c2f32e16b4
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoned0e8598c62dcb5f059a5c2b541c23b09
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone34affb50a3d55c5302e1c7f97aec9679
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoneb3635522e8f89bbe9e56ab7e7f4693fc
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone7389ad9bae0ad1181b7da9e4dd39449d
57 Pipelines and composite estimators 927

scikitlearn user guide Release 0213

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoneab59173e00ab4a7bdf7740e5c6d50123

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone42894538fd0ab2b4235e55ef53b9cd59

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone580596a10617556e39d5e069c4de096a

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone591acae8e60c9632aa990e4fa0962a67

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone323909961fcc2a4950defb581f08007f

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone3654e1d61587aee4f9980c99541d55d3

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone835820e75be3172b4e2e257028147bb2

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone90604ce8bb756e3ff1acd4c1ce065b1a

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone3c66df347f488dfe044ecf991ca2498b

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformonebf4bec5fe5245bf3c0d271e31bc2342f

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformonece0f911df93a8e38932c48f983e7fde6

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone285efba3f9e5b1a35d6825e37a718019

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone74083d6e1ed743fa1e756a41ba467e6c

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoneb72f745822a406f2191b6ebfd8ca9df9

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone997e4ce30ba5143e90330db664bf61c5

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone3140177f6ccbe72992772342347a166f

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformonef843896ad42a20c4a50c0fd29512111a

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoned91a2d083b8c7311c87edd6592fcd446

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone50eb0f4152cd0f7dce3b471add31d07d

fittransformone cache loaded 00s 00min

928 Chapter 5 Examples

scikitlearn user guide Release 0213

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformone3b28284b39c934cb519f2a31e03a951b
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformoneaded0eb932b9c692c801c6c56dc2bda2
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformonecdf3d1530a8a2388ba80a16c66413409
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformoneeff5666a0cb943711a4e4dfd5b5cd8587
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformone536a5fbb5a6c8d69b4fb426f0b51d9eb
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformone26cd3a8242acb986f98cf38291ef5baa
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformone8b6094f421dab9fc88fb54c2f32e16b4
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformoned0e8598c62dcb5f059a5c2b541c23b09
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformone34affb50a3d55c5302e1c7f97aec9679
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformoneb3635522e8f89bbe9e56ab7e7f4693fc
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformone7389ad9bae0ad1181b7da9e4dd39449d
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformoneab59173e00ab4a7bdf7740e5c6d50123
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformone42894538fd0ab2b4235e55ef53b9cd59
fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↳sklearnpipelinefittransformone580596a10617556e39d5e069c4de096a
fittransformone cache loaded 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk2 scorefuncfunction chi2 at 0x7efe30bb2268
↳array0 0

0 0 array0 8 None messageclsnamePipeline
↳messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk2 scorefuncfunction chi2 at 0x7efe30bb2268
↳array0 0

0 0 array0 8 None messageclsnamePipeline
↳messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk2 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk2 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk2 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 9 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk4 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk4 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk4 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk4 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
930 Chapter 5 Examples

scikitlearn user guide Release 0213
fittransformoneSelectKBestk4 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 9 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk8 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk8 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk8 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk8 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 8 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformoneSelectKBestk8 scorefuncfunction chi2 at 0x7efe30bb2268
↪array0 0

0 0 array0 9 None messageclsnamePipeline
↪messageNone
fittransformone 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone0c252f8c2af87fe4a595ca853ea983cd
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone5feb279acc70b729ede7514b133d0172
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone6742bad9e26dd16b831fc9abd1883c9e
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone096380a7ad90487199a4a6e8ec8a5744
fittransformone cache loaded 00s 00min
57 Pipelines and composite estimators 931

scikitlearn user guide Release 0213

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonee5107ef0d6468f91a5cd772a596ba876
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone2efa56788b791becec2fcd015ecf751f
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoned27c6056fc9ab1f3fb4327fd17346312
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoned649c35672f6e2731789f5c6a9b03066
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone4906c4afa619398fb77288e086d5dd82
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone7dc3c29273fd5c57e2913f82c9185294
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone63c1c1af7115994b5cd17a1189691d5b
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone670746700924677f7a550fa4cae5c9bb
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonef9617292c7b763d23b3f6c9d96697c29
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone9a2bcb4639d1b192e5525c2bf0dca317
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone87c7c41885d63769a203e26b244fd823
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone0c252f8c2af87fe4a595ca853ea983cd
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone5feb279acc70b729ede7514b133d0172
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone6742bad9e26dd16b831fc9abd1883c9e
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone096380a7ad90487199a4a6e8ec8a5744
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformonee5107ef0d6468f91a5cd772a596ba876
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone2efa56788b791becec2fcd015ecf751f
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoned27c6056fc9ab1f3fb4327fd17346312
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformoned649c35672f6e2731789f5c6a9b03066
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone4906c4afa619398fb77288e086d5dd82
932 Chapter 5 Examples

scikitlearn user guide Release 0213

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone7dc3c29273fd5c57e2913f82c9185294

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone63c1c1af7115994b5cd17a1189691d5b

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone670746700924677f7a550fa4cae5c9bb

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformonef9617292c7b763d23b3f6c9d96697c29

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone9a2bcb4639d1b192e5525c2bf0dca317

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone87c7c41885d63769a203e26b244fd823

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone0c252f8c2af87fe4a595ca853ea983cd

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone5feb279acc70b729ede7514b133d0172

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone6742bad9e26dd16b831fc9abd1883c9e

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone096380a7ad90487199a4a6e8ec8a5744

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformonee5107ef0d6468f91a5cd772a596ba876

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone2efa56788b791becec2fcd015ecf751f

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoned27c6056fc9ab1f3fb4327fd17346312

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformoned649c35672f6e2731789f5c6a9b03066

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone4906c4afa619398fb77288e086d5dd82

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone7dc3c29273fd5c57e2913f82c9185294

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone63c1c1af7115994b5cd17a1189691d5b

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformone670746700924677f7a550fa4cae5c9bb

fittransformone cache loaded 00s 00min

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib

↪sklearnpipelinefittransformonef9617292c7b763d23b3f6c9d96697c29

fittransformone cache loaded 00s 00min

57 Pipelines and composite estimators 933

scikitlearn user guide Release 0213

Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone9a2bcb4639d1b192e5525c2bf0dca317
fittransformone cache loaded 00s 00min
Memory00s 00min Loading fittransformone from tmptmpl5ribdvujoblib
↪sklearnpipelinefittransformone87c7c41885d63769a203e26b244fd823
fittransformone cache loaded 00s 00min

Memory Calling sklearnpipelinefittransformone
fittransformonePCAiteratedpower7 ncomponents8 array0 0

0 0 array0 8 None messageclsnakePipeline
↪messageNone
fittransformone 00s 00min

ThePCA fitting is only computed at the evaluation of the first configuration of the Cparameter of the LinearSVC classifier The other configurations of Cwill trigger the loading of the cached PCA (Principal Component Analysis) estimator data leading to save processing time Therefore the use of caching the pipeline using memory is highly beneficial when fitting a transformer is costly

Total running time of the script 0 minutes 20695 seconds

Note Click here to download the full example code

575 Column Transformer with Heterogeneous Data Sources

Datasets can often contain components of that require different feature extraction and processing pipelines This scenario might occur when

1 Your dataset consists of heterogeneous data types eg raster images and text captions

2 Your dataset is stored in a Pandas DataFrame and different columns require different processing pipelines

This example demonstrates how to use sklearncomposeColumnTransformer on a dataset containing different types of features We use the 20newsgroups dataset and compute standard bagofwords features for the subject line and body in separate pipelines as well as ad hoc features on the body We combine them with weights using a ColumnTransformer and finally train a classifier on the combined set of features

The choice of features is not particularly helpful but serves to illustrate the technique
Out

Pipeline step 1 of 3 Processing subjectbody total 01s

Pipeline step 2 of 3 Processing union total 03s

Pipeline step 3 of 3 Processing svc total 03s

precision recall f1score support

0 096 062 076 494

1 025 084 039 76

accuracy 065 570

macro avg 061 073 057 570

weighted avg 087 065 071 570

934 Chapter 5 Examples

scikitlearn user guide Release 0213
Author Matt Terry mattterrygmailcom

```
License BSD (Binary Search Decision) 3 clause
import numpy as np
from sklearnbase import BaseEstimator TransformerMixin
from sklearndatasets import fetch20newsgroups
from sklearndatasetstwentynewsgroups import stripnewsgroupfooter
from sklearndatasetstwentynewsgroups import stripnewsgroupquoting
from sklearndecomposition import TruncatedSVD
from sklearnfeatureextraction import DictVectorizer
from sklearnfeatureextractiontext import TfidfVectorizer
from sklearnmetrics import classificationreport
from sklearnpipeline import Pipeline
from sklearncompose import ColumnTransformer
from sklearnsvm import LinearSVC
class TextStats BaseEstimator TransformerMixin
Extract features from each document for DictVectorizer
deffitself x yNone
returnself
deftransformself posts
returnlength lentext
numsentences textcount
fortextinposts
class SubjectBodyExtractor BaseEstimator TransformerMixin
Extract the subject body from a usenet post in a single pass
Takes a sequence of strings and produces a dict of sequences Keys are
subject and body

deffitself x yNone
returnself
deftransformself posts
construct object dtype array with two columns
first column subject and second column body
features npemptyshapelenposts 2 dtypeobject
fori text inenumerateposts
headers bod textpartition nn
bod stripnewsgroupfooterbod
bod stripnewsgroupquotingbod
featuresi 1 bod
prefix Subject
sub
forlineinheaderssplit n
iflinestartswithprefix
sub linelenprefix
break
featuresi 0 sub
57 Pipelines and composite estimators 935
```

```
scikitlearn user guide Release 0213
returnfeatures
pipeline Pipeline
    Extract the subject body
subjectbody SubjectBodyExtractor
    Use ColumnTransformer to combine the features from subject and body
union ColumnTransformer

    Pulling features from the posts subject line first column
subject TfidfVectorizermindf50 0
    Pipeline for standard bagofwords model for body second column
bodybow Pipeline
tfidf TfidfVectorizer
best TruncatedSVDncomponents50
1
    Pipeline for pulling ad hoc features from posts body
bodystats Pipeline
stats TextStats returns a list of dicts
vect DictVectorizer list of dicts feature matrix
1

    weight components in ColumnTransformer
transformerweights
subject 08
bodybow 05
bodystats 10

    Use a SVC (Support Vector Classification) classifier on the combined features
svc LinearSVC
verboseTrue
    limit the list of categories to make running this example faster
categories altatheism talkreligionmisc
train fetch20newsgroupsrandomstate1
subsettrain
categoriescategories

test fetch20newsgroupsrandomstate1
subsettest
categoriescategories

pipelinefittraindata traintarget
y pipelinepredicttestdata
printclassificationreporty testtarget
Total running time of the script 0 minutes 1258 seconds
Note Click here to download the full example code
936 Chapter 5 Examples
```

scikitlearn user guide Release 0213

576 Effect of transforming the targets in regression model

In this example we give an overview of the `sklearn.compose.TransformedTargetRegressor`. Two examples illustrate the benefit of transforming the targets before learning a linear regression model. The first example uses synthetic data while the second example is based on the Boston housing data set.

Author: Guillaume Lemaître, [guillaumelemaître@nria.fr](#)

License: BSD (Binary Search Decision) 3 clause

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from distutils.version import LooseVersion
printdoc
Synthetic example
from sklearn.datasets import make_regression
from sklearn.model_selection import train_test_split
from sklearn.linear_model import RidgeCV
from sklearn.compose import TransformedTargetRegressor
from sklearn.metrics import median_absolute_error, r2_score
# normed is being deprecated in favor of density in histograms
if LooseVersion(matplotlib.__version__) < 2.1:
    density_param = density = True
else:
    density_param = normed = True
```

A synthetic random regression problem is generated. The targets y are modified by i translating all targets such that all entries are nonnegative and ii applying an exponential function to obtain nonlinear targets which cannot be fitted using a simple linear model.

Therefore a logarithmic `np.log1p` and an exponential function `np.exp(m1)` will be used to transform the targets before training a linear regression model and using it for prediction.

```
X, y = make_regression(n_samples=10000, noise=100, random_state=0)
y = npexp(y) - abs(y.min())
y_trans = nplog1p(y)
```

The following illustrate the probability density functions of the target before and after applying the logarithmic functions.

```
fig, (ax0, ax1) = plt.subplots(1, 2)
ax0.hist(y, bins=100, density_param)
ax0.set_xlim(0, 2000)
ax0.set_ylabel('Probability')
ax0.set_xlabel('Target')
ax0.set_title('Target distribution')
ax1.hist(y_trans, bins=100, density_param)
ax1.set_ylabel('Probability')
```

```
scikitlearn user guide Release 0213
ax1setxlabelTarget
ax1settitleTransformed target distribution
fsuptitleSynthetic data y0035
ftightlayoutrect005 005 095 095
Xtrain Xtest ytrain ytest traintestsplitX y randomstate0
At first a linear model will be applied on the original targets Due to the nonlinearity the model trained will not
be precise during the prediction Subsequently a logarithmic function is used to linearize the targets allowing better
prediction even with a similar linear model as reported by the median absolute error MAE
f ax0 ax1 pltsubplots1 2 shareyTrue
regr RidgeCV
regrfitXtrain ytrain
ypred regrpredictXtest
ax0scatterytest ypred
ax0plot0 2000 0 2000 k
ax0setylabelTarget predicted
ax0setxlabelTrue Target
ax0settitleRidge regression nwithout target transformation
ax0text100 1750 rR2 2f MAE2f
r2scoreytest ypred medianabsoluteerror ytest ypred
ax0setxlim0 2000
938 Chapter 5 Examples
```


scikitlearn user guide Release 0213
ax0setylim0 2000
regrtrans TransformedTargetRegressorregressorRidgeCV
funcnplog1p
inversefuncnpexpm1
regrtransfitXtrain ytrain
ypred regrtranspredictXtest
ax1scatterytest ypred
ax1plot0 2000 0 2000 k
ax1setylabelTarget predicted
ax1setxlabelTrue Target
ax1settitleRidge regression nwith target transformation
ax1text100 1750 rR2 2f MAE2f
r2scoreytest ypred medianabsoluteerrorrytest ypred
ax1setxlim0 2000
ax1setylim0 2000
fsuptitleSynthetic data y0035
ftightlayoutrect005 005 095 095
57 Pipelines and composite estimators 939

scikitlearn user guide Release 0213

Realworld data set

In a similar manner the boston housing data set is used to show the impact of transforming the targets before learning a model In this example the targets to be predicted corresponds to the weighted distances to the five Boston employment centers

```
from sklearn.datasets import load_boston
from sklearn.preprocessing import QuantileTransformer, quantile_transform
dataset = load_boston
target = np.array(dataset.feature_names == 'DIS')
X = dataset.data[:, np.logical_not(target)]
y = dataset.data[target].squeeze()
ytrans = quantile_transform(dataset.data[target],
                             nquantiles=300,
                             output_distribution='normal',
                             copy=True).squeeze()
# sklearn.preprocessing.QuantileTransformer is used such that the targets follows a normal distribution before applying a sklearn.linear_model.RidgeCV model
fig, (ax0, ax1) = plt.subplots(1, 2)
ax0.hist(y, bins=100, density=True)
ax0.set_ylabel('Probability')
ax0.set_xlabel('Target')
ax0.set_title('Target distribution')
ax1.hist(ytrans, bins=100, density=True)
ax1.set_ylabel('Probability')
ax1.set_xlabel('Target')
ax1.set_title('Transformed target distribution')
fig.suptitle('Boston housing data distance to employment centers')
fig.tight_layout()
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)
```

scikitlearn user guide Release 0213

The effect of the transformer is weaker than on the synthetic data However the transform induces a decrease of the MAE

f ax0 ax1 pltsubplots1 2 shareyTrue

regr RidgeCV

regrfitXtrain ytrain

ypred regrpredictXtest

ax0scatterytest ypred

ax0plot0 10 0 10 k

ax0setylabelTarget predicted

ax0setxlabelTrue Target

ax0settitleRidge regression nwithout target transformation

ax0text1 9 rR2 2f MAE2f

r2scoreytest ypred medianabsoluteerrorytest ypred

ax0setxlim0 10

ax0setylim0 10

regrtrans TransformedTargetRegressor

regressorRidgeCV

transformerQuantileTransformernquantiles300

outputdistributionnormal

regrtransfitXtrain ytrain

ypred regrtranspredictXtest

57 Pipelines and composite estimators 941

scikitlearn user guide Release 0213
ax1scatterypred ypred
ax1plot0 10 0 10 k
ax1setylabelTarget predicted
ax1setxlabelTrue Target
ax1settitleRidge regression nwith target transformation
ax1text1 9 rR2 2f MAE2f
r2scoreypred ypred medianabsoluteerrorypred ypred
ax1setxlim0 10
ax1setylim0 10
fsuptitleBoston housing data distance to employment centers y0035
ftightlayoutrect005 005 095 095
pltshow
Total running time of the script 0 minutes 1682 seconds
58 Covariance estimation
Examples concerning the sklearn covariance module
Note Click here to download the full example code
942 Chapter 5 Examples

scikitlearn user guide Release 0213

581 LedoitWolf vs OAS estimation

The usual covariance maximum likelihood estimate can be regularized using shrinkage Ledoit and Wolf proposed a close formula to compute the asymptotically optimal shrinkage parameter minimizing a MSE criterion yielding the LedoitWolf covariance estimate

Chen et al proposed an improvement of the LedoitWolf shrinkage parameter the OAS coefficient whose convergence is significantly better under the assumption that the data are Gaussian

This example inspired from Chen’s publication 1 shows a comparison of the estimated MSE of the LW and OAS methods using Gaussian distributed data

1 “Shrinkage Algorithms for MMSE Covariance Estimation” Chen et al IEEE Trans on Sign Proc V olume 58 Issue 10 October 2010

```
printdoc
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import toeplitz cholesky
from sklearn.covariance import LedoitWolf OAS
np.random.seed(0)
nfeatures = 100
# simulation covariance matrix AR1 process
r = 0.1
realcov = toeplitz(r * np.arange(nfeatures))
coloringmatrix = cholesky(realcov)
nsamplesrange = np.arange(6, 31, 1)
repeat = 100
lwmse = np.zeros(nsamplesrange.size, repeat)
oamse = np.zeros(nsamplesrange.size, repeat)
lwshrinkage = np.zeros(nsamplesrange.size, repeat)
oashrinkage = np.zeros(nsamplesrange.size, repeat)
for i, nsamples in enumerate(nsamplesrange):
    for j in range(repeat):
        X = np.dot(
            np.random.randn(nsamples, nfeatures),
            coloringmatrix.T
        )
        lw = LedoitWolf(store_precision=False, assume_centered=True)
        lw.fit(X)
        lwmse[j, i] = lw.error_norm(realcov, scaling=False)
        lwshrinkage[j, i] = lw.shrinkage
        oa = OAS(store_precision=False, assume_centered=True)
        oa.fit(X)
        oamse[j, i] = oa.error_norm(realcov, scaling=False)
        oashrinkage[j, i] = oa.shrinkage
    plot_MSE
    plt.subplot(2, 1, 1)
    plt.errorbar(nsamplesrange, lwmse.mean(1), yerr=lwmse.std(1),
        label=LedoitWolf, color=navy, lw=2)
    plt.errorbar(nsamplesrange, oamse.mean(1), yerr=oamse.std(1),
        label=OAS, color=darkorange, lw=2)
    plt.ylabel('Squared error')
58 Covariance estimation 943
```

```
scikitlearn user guide Release 0213
pltlegendlocupper right
plttitleComparison of covariance estimators
pltxlim5 31
plot shrinkage coefficient
pltsubplot2 1 2
plterrorbarsamplesrange lwshrinkagemean1 yerrlwshrinkagestd1
labelLedoitWolf colornavy lw2
plterrorbarsamplesrange oashrinkagemean1 yerroashrinkagestd1
labelOAS colordarkorange lw2
pltxlabelnsamples
pltylabelShrinkage
pltlegendloclower right
pltylimpltylim0 1 pltylim1 pltylim0 10
pltxlim5 31
pltshow
Total running time of the script 0 minutes 3818 seconds
Note Click here to download the full example code
944 Chapter 5 Examples
```

582 Sparse inverse covariance estimation

Using the GraphicalLasso estimator to learn a covariance and sparse precision from a small number of samples
To estimate a probabilistic model eg a Gaussian model estimating the precision matrix that is the inverse covariance matrix is as important as estimating the covariance matrix Indeed a Gaussian model is parametrized by the precision matrix

To be in favorable recovery conditions we sample the data from a model with a sparse inverse covariance matrix In addition we ensure that the data is not too much correlated limiting the largest coefficient of the precision matrix and that there are no small coefficients in the precision matrix that cannot be recovered In addition with a small number of observations it is easier to recover a correlation matrix rather than a covariance thus we scale the time series Here the number of samples is slightly larger than the number of dimensions thus the empirical covariance is still invertible However as the observations are strongly correlated the empirical covariance matrix is illconditioned and as a result its inverse –the empirical precision matrix– is very far from the ground truth

If we use l_2 shrinkage as with the LedoitWolf estimator as the number of samples is small we need to shrink a lot As a result the LedoitWolf precision is fairly close to the ground truth precision that is not far from being diagonal but the offdiagonal structure is lost

The l_1 penalized estimator can recover part of this offdiagonal structure It learns a sparse precision It is not able to recover the exact sparsity pattern it detects too many nonzero coefficients However the highest nonzero coefficients of the l_1 estimated correspond to the nonzero coefficients in the ground truth Finally the coefficients of the l_1 precision estimate are biased toward zero because of the penalty they are all smaller than the corresponding ground truth value as can be seen on the figure

Note that the color range of the precision matrices is tweaked to improve readability of the figure The full range of values of the empirical precision is not displayed

The alpha parameter of the GraphicalLasso setting the sparsity of the model is set by internal crossvalidation in the GraphicalLassoCV As can be seen on figure 2 the grid to compute the crossvalidation score is iteratively refined in the neighborhood of the maximum

scikitlearn user guide Release 0213

```
•
printdoc
author Gael Varoquaux gaelvaroquauxinriafr
License BSD (Binary Search Decision) 3 clause
Copyright INRIA
import numpy as np
from scipy import linalg
from sklearn.datasets import make_sparse_positive_matrix
from sklearn.covariance import GraphicalLassoCV, LedoitWolf
import matplotlib.pyplot as plt
```

```

Generate the data
nsamples = 60
nfeatures = 20
prng = np.random.RandomState(1)
prec = make_sparse_positive_matrix(nfeatures, nfeatures, alpha=98)
smallest_coef = 4
largest_coef = 7
random_state = prng
cov = linalg.pinv(prec)
d = np.sqrt(np.diag(cov))
cov = d * np.eye(nfeatures)
prec = d * np.eye(nfeatures)
X = prng.multivariate_normal(np.zeros(nfeatures), cov, size=nsamples)
X_mean = X.mean(axis=0)
X_std = X.std(axis=0)
```

```

Estimate the covariance
emp_cov = np.dot(X.T, X) / nsamples
model = GraphicalLassoCV(cv=5)
model.fit(X)
cov = model.covariance_
946 Chapter 5 Examples
```


scikitlearn user guide Release 0213

prec modelprecision
lwcov ledoitwolfX
lwprec linalginvlwcov

Plot the results
pltfigurefigsize10 6
pltsubplotsadjustleft002 right098
plot the covariances
covs Empirical empcov LedoitWolf lwcov
GraphicalLassoCV cov True cov
vmax covmax
fori name thiscov inenumeratecovs
pltsubplot2 4 i 1
pltimshowthiscov interpolationnearest vminvmax vmaxvmax
cmappltcmRdBur
pltxticks
pltyticks
plttitle scovariance name
plot the precisions
prec Empirical linalginvempcov LedoitWolf lwprec
GraphicalLasso prec True prec
vmax 9 precmax
fori name thisprec inenumerateprec
ax pltsubplot2 4 i 5
pltimshownpmmaskedequalthisprec 0
interpolationnearest vminvmax vmaxvmax
cmappltcmRdBur
pltxticks
pltyticks
plttitle sprecision name
ifhasattr setfacecolor
axsetfacecolor7
else
axsetaxisbgcolor7
plot the model selection metric
pltfigurefigsize4 3
pltaxes2 15 75 7
pltplotmodelcvalphas npmeanmodelgridscores axis1 o
pltaxvlinemodelalpha color5
plttitleModel selection
pltlabelCrossvalidation score
pltlabelalpha
pltshow
Total running time of the script 0 minutes 0501 seconds
Note [Click here to download the full example code](#)
58 Covariance estimation 947

583 Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood

When working with covariance estimation the usual approach is to use a maximum likelihood estimator such as `sklearn.covariance.EmpiricalCovariance`. It is unbiased ie it converges to the true population covariance when given many observations. However it can also be beneficial to regularize it in order to reduce its variance this in turn introduces some bias. This example illustrates the simple regularization used in `ShrunkCovariance` estimators. In particular it focuses on how to set the amount of regularization ie how to choose the bias-variance tradeoff.

Here we compare 3 approaches:

- Setting the parameter by crossvalidating the likelihood on three folds according to a grid of potential shrinkage parameters
- A close formula proposed by Ledoit and Wolf to compute the asymptotically optimal regularization parameter minimizing a MSE criterion yielding the `sklearn.covariance.LedoitWolf` covariance estimate
- An improvement of the LedoitWolf shrinkage the `sklearn.covariance.OAS` proposed by Chen et al

Its convergence is significantly better under the assumption that the data are Gaussian in particular for small samples.

To quantify estimation error we plot the likelihood of unseen data for different values of the shrinkage parameter. We also show the choices by crossvalidation or with the LedoitWolf and OAS estimates.

Note that the maximum likelihood estimate corresponds to no shrinkage and thus performs poorly. The LedoitWolf estimate performs really well as it is close to the optimal and is computationally not costly. In this example the OAS estimate is a bit further away. Interestingly both approaches outperform crossvalidation which is significantly most computationally costly.

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from scipy import linalg
from sklearn.covariance import LedoitWolf OAS ShrunkCovariance
loglikelihood empiricalcovariance
from sklearn.model_selection import GridSearchCV
```

```
Generate sample data
nfeatures nsamples 40 20
np.random.seed(42)
baseXtrain np.random.randn(samples nfeatures)
baseXtest np.random.randn(samples nfeatures)
Color samples
coloringmatrix np.random.randn(samples nfeatures)
Xtrain np.dot(baseXtrain coloringmatrix)
Xtest np.dot(baseXtest coloringmatrix)
```

```
Compute the likelihood on test data
58 Covariance estimation 949
```

scikitlearn user guide Release 0213
spanning a range of possible shrinkage coefficient values
shrinkages nplogspace2 0 30
negativelogliks ShrunkCovarianceshrinkagesfitXtrainscoreXtest
forsinshrinkages
under the groundtruth model which we would not have access to in real
settings
realcov npdotcoloringmatrixT coloringmatrix
empcov empiricalcovarianceXtrain
loglikreal loglikelihoodempcov linalginvrealcov

Compare different approaches to setting the parameter
GridSearch for an optimal shrinkage coefficient
tunedparameters shrinkage shrinkages
cv GridSearchCVShrunkCovariance tunedparameters cv5
cvfitXtrain
LedoitWolf optimal shrinkage coefficient estimate
lw LedoitWolf
logliklw lwfitXtrainscoreXtest
OAS coefficient estimate
oa OAS
loglikoa oafitXtrainscoreXtest

Plot results
fig pltfigure
plttitleRegularized covariance likelihood and shrinkage coefficient
pltxlabelRegularization parameter shrinkage coefficient
pltylabelError negative loglikelihood on test data
range shrinkage curve
pltloglogshrinkages negativelogliks labelNegative loglikelihood
pltplotpltxlim 2 loglikreal r
labelReal covariance likelihood
adjust view
likmax npamaxnegativelogliks
likmin npaminnegativelogliks
ymin likmin 6 nplogpltylim1 pltylim0
ymax likmax 10 nploglikmax likmin
xmin shrinkages0
xmax shrinkages1
LW likelihood
pltvlineslwshrinkage ymin logliklw colormagenta
linewidth3 labelLedoitWolf estimate
OAS likelihood
pltvlinesoashrinkage ymin loglikoa colorpurple
linewidth3 labelOAS estimate
best CV (Cross-Validation) estimator likelihood
pltvlinescvbestestimatorshrinkage ymin
cvbestestimatorscoreXtest colorcyan
linewidth3 labelCrossvalidation best estimate
pltylimymin ymax
950 Chapter 5 Examples

scikitlearn user guide Release 0213

pltxlimxmin xmax

pltlegend

pltshow

Total running time of the script 0 minutes 0183 seconds

Note Click here to download the full example code

584 Robust covariance estimation and Mahalanobis distances relevance

An example to show covariance estimation with the Mahalanobis distances on Gaussian distributed data

For Gaussian distributed data the distance of an observation \mathbf{x} to the mode of the distribution can be computed using its Mahalanobis distance $\sqrt{(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$ where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the location and the covariance of the underlying Gaussian distribution

In practice $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are replaced by some estimates The usual covariance maximum likelihood estimate is very sensitive to the presence of outliers in the data set and therefor the corresponding Mahalanobis distances are One would better have to use a robust estimator of covariance to guarantee that the estimation is resistant to “erroneous” observations in the data set and that the associated Mahalanobis distances accurately reflect the true organisation of the observations

The Minimum Covariance Determinant estimator is a robust highbreakdown point ie it can be used to estimate the covariance matrix of highly contaminated datasets up to $\frac{n-p}{2}$ samples – $\frac{p}{2}$ features – 1

2 outliers estimator of covariance The idea is

to find $\frac{n-p}{2}$ samples $\frac{p}{2}$ features 1

2 observations whose empirical covariance has the smallest determinant yielding a “pure” subset

of observations from which to compute standards estimates of location and covariance

The Minimum Covariance Determinant estimator MCD has been introduced by P J Rousseeuw in 1

This example illustrates how the Mahalanobis distances are affected by outlying data observations drawn from a contaminating distribution are not distinguishable from the observations coming from the real Gaussian distribution

that one may want to work with Using MCD based Mahalanobis distances the two populations become distinguish

able Associated applications are outliers detection observations ranking clustering For visualization purpose

the cubic root of the Mahalanobis distances are represented in the boxplot as Wilson and Hilferty suggest 2

1 P J Rousseeuw Least median of squares regression J Am Stat Ass 79 871 1984

2 Wilson E B Hilferty M M 1931 The distribution of chisquare Proceedings of the National

Academy of Sciences of the United States of America 17 684 688

58 Covariance estimation 951

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.covariance import EmpiricalCovariance MinCovDet
nsamples 125
noutliers 25
nfeatures 2
generate data
gencov npeyenfeatures
gencov0 0 2
X npdotnprandomrandnnnsamples nfeatures gencov
add some outliers
outlierscov npeyenfeatures
outlierscovnparange1 nfeatures nparange1 nfeatures 7
Xnoutliers npdotnprandomrandnnnoutliers nfeatures outlierscov
fit a Minimum Covariance Determinant MCD robust estimator to data
robustcov MinCovDetfitX
compare estimators learnt from the full data set with true parameters
empcov EmpiricalCovariancefitX
952 Chapter 5 Examples
```

```

    Display results
fig pltfigure
pltsubplotsadjustspace1 wspace4 top95 bottom05
    Show data set
subfig1 pltsubplot3 1 1
inlierplot subfig1scatterX 0 X 1
colorblack labelinliers
outlierplot subfig1scatterX 0noutliers X 1noutliers
colorred labeloutliers
subfig1setxlimsubfig1getxlim0 11
subfig1settitleMahalanobis distances of a contaminated data set
    Show contours of the distance functions
xx yy npmeshgridnplinspacepltxlim0 pltxlim1 100
nplinspaceptylim0 ptylim1 100
zz npcxxravel yy.ravel
mahalempcov empcovmahalanobiszz
mahalempcov mahalempcovreshapexxshape
empcovcontour subfig1contourxx yy npsqrtmahalempcov
cmapppltcmPuBur
linestylesdashed
mahalrobustcov robustcovmahalanobiszz
mahalrobustcov mahalrobustcovreshapexxshape
robustcontour subfig1contourxx yy npsqrtmahalrobustcov
cmapppltcmYIOrBrr linestylesdotted
subfig1legendempcovcontourcollections1 robustcontourcollections1
inlierplot outlierplot
MLE dist robust dist inliers outliers
locupper right borderaxespad0
pltxticks
ptyticks
    Plot the scores for each point
empmahal empcovmahalanobisX npmeanX 0 033
subfig2 pltsubplot2 2 3
subfig2boxplotempmahalnoutliers empmahalnoutliers widths25
subfig2plotnpfullnsamples noutliers 126
empmahalnoutliers k markeredgewidth1
subfig2plotnpfullnoutliers 226
empmahalnoutliers k markeredgewidth1
subfig2axessetxticklabelsinliers outliers size15
subfig2setylabelsqrt3rmMahal dist size16
subfig2settitle1 from nonrobust estimates nMaximum Likelihood
ptyticks
robustmahal robustcovmahalanobisX robustcovlocation 033
subfig3 pltsubplot2 2 4
subfig3boxplotrobustmahalnoutliers robustmahalnoutliers
widths25
subfig3plotnpfullnsamples noutliers 126
robustmahalnoutliers k markeredgewidth1
subfig3plotnpfullnoutliers 226
robustmahalnoutliers k markeredgewidth1
58 Covariance estimation 953
```

scikitlearn user guide Release 0213
subfig3axessetxticklabelsinliers outliers size15
subfig3setylabelsqrt3rmMahal dist size16
subfig3settitle2 from robust estimates nMinimum Covariance Determinant
plt.xticks
plt.show
Total running time of the script 0 minutes 0298 seconds
Note Click here to download the full example code
585 Robust vs Empirical covariance estimate
The usual covariance maximum likelihood estimate is very sensitive to the presence of outliers in the data set In such a case it would be better to use a robust estimator of covariance to guarantee that the estimation is resistant to “erroneous” observations in the data set12
Minimum Covariance Determinant Estimator
The Minimum Covariance Determinant estimator is a robust highbreakdown point ie it can be used to estimate the covariance matrix of highly contaminated datasets up to $\frac{n - \sqrt{n}}{2}$ samples – $\frac{p - \sqrt{p}}{2}$ features – 1
2outliers estimator of covariance The idea is to find $\frac{n - \sqrt{n}}{2}$ samples $\frac{p - \sqrt{p}}{2}$ features1
2observations whose empirical covariance has the smallest determinant yielding a “pure” subset of observations from which to compute standard estimates of location and covariance After a correction step aiming at compensating the fact that the estimates were learned from only a portion of the initial data we end up with robust estimates of the data set location and covariance
The Minimum Covariance Determinant estimator MCD has been introduced by PJRousseeuw in3
Evaluation
In this example we compare the estimation errors that are made when using various types of location and covariance estimates on contaminated Gaussian distributed data sets
• The mean and the empirical covariance of the full dataset which break down as soon as there are outliers in the data set
• The robust MCD that has a low error provided $\frac{n - \sqrt{n}}{2}$ samples 5 $\frac{p - \sqrt{p}}{2}$ features
• The mean and the empirical covariance of the observations that are known to be good ones This can be considered as a “perfect” MCD estimation so one can trust our implementation by comparing to this case
1Johanna Hardin David M Rocke The distribution of robust distances Journal of Computational and Graphical Statistics December 1 2005 144 928946
2Zoubir A Koivunen V Chakhchoukh Y and Muma M 2012 Robust estimation in signal processing A tutorialstyle treatment of fundamental concepts IEEE Signal Processing Magazine 29 4 6180
3P J Rousseeuw Least median of squares regression Journal of American Statistical Ass 79 871 1984
954 Chapter 5 Examples


```
scikitlearn user guide Release 0213
References
printdoc
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.fontmanager
from sklearn.covariance import EmpiricalCovariance MinCovDet
example settings
nsamples 80
nfeatures 5
repeat 10
rangenoutliers np.concatenate
nplinspace0 nsamples 8 5
nplinspace0 nsamples 8 nsamples 2 511 astypenpint
definition of arrays to store results
errlocmcd npzerosrangenoutlierssize repeat
errcovmcd npzerosrangenoutlierssize repeat
errlocempfull npzerosrangenoutlierssize repeat
errcovempfull npzerosrangenoutlierssize repeat
errlocemppure npzerosrangenoutlierssize repeat
errcovemppure npzerosrangenoutlierssize repeat
58 Covariance estimation 955
```

```
scikitlearn user guide Release 0213
computation
fori noutliers inenumeratorangenoutliers
forjinrangerepeat
rng nprandomRandomStatei j
generate data
X rnggrandnnsamples nfeatures
add some outliers
outliersindex rngpermutationnnsamplesnoutliers
outliersoffset 10
nprandomrandint2 sizednoutliers nfeatures 05
Xoutliersindex outliersoffset
inliersmask nponesnsamplesastypebool
inliersmaskoutliersindex False
fit a Minimum Covariance Determinant MCD robust estimator to data
mcd MinCovDetfitX
compare raw robust estimates with the true location and covariance
errlocmcdi j npsummcdlocation 2
errcovmcdi j mcderrornormnpeyenfeatures
compare estimators learned from the full data set with true
parameters
errlocempfulli j npsumXmean0 2
errcovempfulli j EmpiricalCovariancefitXerrornorm
npeyenfeatures
compare with an empirical covariance learned from a pure data set
ie perfect mcd
pureX Xinliersmask
purelocation pureXmean0
pureempcov EmpiricalCovariancefitpureX
errlocemppurei j npsumpurelocation 2
errcovemppurei j pureempcoverrornormnpeyenfeatures
Display results
fontprop matplotlibfontmanagerFontPropertiessize11
pltsubplot2 1 1
lw 2
plterrorbarrangenoutliers errlocmcdmean1
yerrerrlocmcdstd1 npqrtrepeat
labelRobust location lwlw colorm
plterrorbarrangenoutliers errlocempfullmean1
yerrerrlocempfullstd1 npqrtrepeat
labelFull data set mean lwlw colorgreen
plterrorbarrangenoutliers errlocemppuremean1
yerrerrlocemppurestd1 npqrtrepeat
labelPure data set mean lwlw colorblack
plttitleInfluence of outliers on the location estimation
pltlabelrError mu hatmu22
pltlegendlocupper left propfontprop
pltsubplot2 1 2
xsize rangenoutlierssize
plterrorbarrangenoutliers errcovmcdmean1
yerrerrcovmcdstd1
956 Chapter 5 Examples
```

scikitlearn user guide Release 0213
labelRobust covariance mcd colorm
plterrorbarrangenoutliersxsize 5 1
errcovempfullmean1xsize 5 1
yerrerrcovempfullstd1xsize 5 1
labelFull data set empirical covariance colorgreen
pltplotrangenoutliersxsize 5size 2 1
errcovempfullmean1xsize 5size 2 1
colorgreen ls
plterrorbarrangenoutliers errcovemppuremean1
yerrerrcovemppurestd1
labelPure data set empirical covariance colorblack
plttitleInfluence of outliers on the covariance estimation
pltxlabelAmount of contamination
pltylabelRMSE
pltlegendlocupper center propfontprop
pltshow
Total running time of the script 0 minutes 2985 seconds
59 Cross decomposition
Examples concerning the sklearncrossdecomposition module
Note Click here to download the full example code
591 Compare cross decomposition methods
Simple usage of various cross decomposition algorithms PLSCanonical PLSRegression with multivariate re
sponse aka PLS2 PLSRegression with univariate response aka PLS1 CCA
Given 2 multivariate covarying twodimensional datasets X and Y PLS extracts the 'directions of covariance' ie
the components of each datasets that explain the most shared variance between both datasets This is apparent on the
scatterplot matrix display components 1 in dataset X and dataset Y are maximally correlated points lie around the
first diagonal This is also true for components 2 in both dataset however the correlation across datasets for different
components is weak the point cloud is very spherical
59 Cross decomposition 957

scikitlearn user guide Release 0213

Out
CorrX
1 051 007 005
051 1 011 001
007 011 1 049
005 001 049 1
CorrY
1 048 005 003
048 1 004 012
005 004 1 051
003 012 051 1
True B such that $Y = XB$ Err
1 1 1
2 2 2
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
Estimated B
1 1 1
2 2 2
0 0 0
0 0 0
0 0 0
958 Chapter 5 Examples

```
scikitlearn user guide Release 0213
0 0 0
0 0 01
0 0 0
0 0 01
0 0 0
Estimated betas
1
21
0
0
0
0
0
0
0
0
0
0
0
0
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cross_decomposition import PLSCanonical, PLSRegression, CCA

Dataset based latent variables model
n = 500
2 latents vars
I1 = np.random.randn(n)
I2 = np.random.randn(n)
latents = np.array([I1, I2]).T
X = latents @ np.random.randn(4, n).T
Y = latents @ np.random.randn(4, n).T
X_train = X[:n]
Y_train = Y[:n]
X_test = X[n:]
Y_test = Y[n:]
print(CorrX)
print(np.corrcoef(X_train.T, X_test.T))
print(CorrY)
print(np.corrcoef(Y_train.T, Y_test.T))

Canonical symmetric PLS
Transform data

plsca = PLSCanonical(n_components=2)
plsca.fit(X_train, Y_train)
X_trainr, Y_trainr = plsca.transform(X_train, Y_train)
59 Cross decomposition 959
```

scikitlearn user guide Release 0213
Xtestr Ytestr plscatransformXtest Ytest
Scatter plot of scores

1 On diagonal plot X vs Y scores on each components

```
pltfigurefigsize12 8
pltsubplot221
pltscatterXtrainr 0 Ytrainr 0 labeltrain
markero cb s25
pltscatterXtestr 0 Ytestr 0 labeltest
markero cr s25
pltlabelx scores
pltlabely scores
plttitleComp 1 X vs Y test corr 2f
npcorrcoefXtestr 0 Ytestr 00 1
pltxticks
pltxticks
pltlegendlocbest
pltsubplot224
pltscatterXtrainr 1 Ytrainr 1 labeltrain
markero cb s25
pltscatterXtestr 1 Ytestr 1 labeltest
markero cr s25
pltlabelx scores
pltlabely scores
plttitleComp 2 X vs Y test corr 2f
npcorrcoefXtestr 1 Ytestr 10 1
pltxticks
pltxticks
pltlegendlocbest
```

2 Off diagonal plot components 1 vs 2 for X and Y

```
pltsubplot222
pltscatterXtrainr 0 Xtrainr 1 labeltrain
marker cb s50
pltscatterXtestr 0 Xtestr 1 labeltest
marker cr s50
pltlabelX comp 1
pltlabelX comp 2
plttitleX comp 1 vs X comp 2 test corr 2f
npcorrcoefXtestr 0 Xtestr 10 1
pltlegendlocbest
pltxticks
pltxticks
pltsubplot223
pltscatterYtrainr 0 Ytrainr 1 labeltrain
marker cb s50
pltscatterYtestr 0 Ytestr 1 labeltest
marker cr s50
pltlabelY comp 1
pltlabelY comp 2
plttitleY comp 1 vs Y comp 2 test corr 2f
npcorrcoefYtestr 0 Ytestr 10 1
pltlegendlocbest
pltxticks
pltxticks
```

scikitlearn user guide Release 0213
pltshow

PLS regression with multivariate response aka PLS2

```
n 1000
q 3
p 10
X np.random.randn(n,p).reshape(p)
B np.array(1/2 * 0 p 2 qT
each Yj 1 X1 2X2 noise
Y np.dot(X B np.random.randn(n,q).reshape(q 5
pls2 PLSRegression(ncomponents=3
pls2.fit(X Y
print True B such that Y XB Err
print B
compare pls2.coef with B
print Estimated B
print np.round(pls2.coef 1
pls2.predict(X
```

PLS regression with univariate response aka PLS1

```
n 1000
p 10
X np.random.randn(n,p).reshape(p)
y X 0 2 X 1 np.random.randn(n 1 5
pls1 PLSRegression(ncomponents=3
pls1.fit(X y
note that the number of components exceeds 1 the dimension of y
print Estimated betas
print np.round(pls1.coef 1
```

CCA PLS mode B with symmetric deflation

```
cca CCAncomponents=2
cca.fit(Xtrain Ytrain
Xtrainr Ytrainr cca.transform(Xtrain Ytrain
Xtestr Ytestr cca.transform(Xtest Ytest
Total running time of the script 0 minutes 0099 seconds
510 Dataset examples
Examples concerning the sklearn.datasets module
Note Click here to download the full example code
510 Dataset examples 961
```

scikitlearn user guide Release 0213

5101 The Digit Dataset

This dataset is made up of 1797 8x8 images Each image like the one shown below is of a handwritten digit In order to utilize an 8x8 figure like this we'd have to first transform it into a feature vector with length 64

See here for more information about this dataset

printdoc

Code source Gaël Varoquaux

Modified for documentation by Jaques Grobler

License BSD (Binary Search Decision) 3 clause

from sklearn import datasets

import matplotlib.pyplot as plt

Load the digits dataset

digits = datasets.load_digits

Display the first digit

plt.figure(1, figsize=(3, 3))

plt.imshow(digits.images[0], cmap=plt.cm.gray, interpolation='nearest')

plt.show

Total running time of the script: 0 minutes 0113 seconds

Note: Click here to download the full example code

5102 The Iris Dataset

This data set consists of 3 different types of irises: Setosa Versicolour and Virginica petal and sepal length stored in a 150x4 numpy ndarray

The rows being the samples and the columns being Sepal Length Sepal Width Petal Length and Petal Width

962 Chapter 5 Examples

scikitlearn user guide Release 0213
The below plot uses the first two features See here for more information on this dataset
•
510 Dataset examples 963

scikitlearn user guide Release 0213

```
•
printdoc
Code source Gaël Varoquaux
Modified for documentation by Jaques Grobler
License BSD (Binary Search Decision) 3 clause
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn import datasets
from sklearn.decomposition import PCA (Principal Component Analysis)
import some data to play with
iris = datasets.load_iris()
X = iris.data[:, :2] # we only take the first two features
y = iris.target
xmin, xmax = X[:, 0].min(), X[:, 0].max()
ymin, ymax = X[:, 1].min(), X[:, 1].max()
plt.figure(2, figsize=(8, 6))
plt.clf()
# Plot the training points
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1,
            edgecolor='k')
plt.xlabel('Sepal length')
964 Chapter 5 Examples
```

scikitlearn user guide Release 0213

plt.ylabel('Sepal width')

plt.xlim(xmin, xmax)

plt.ylim(ymin, ymax)

plt.xticks

plt.yticks

To get a better understanding of interaction of the dimensions

plot the first three PCA (Principal Component Analysis) dimensions

fig, plt = plt.figure(figsize=(8, 6))

ax = Axes3D(fig, elev=150, azim=110)

X_reduced = PCA(n_components=3).fit_transform(iris.data)

ax.scatter(X_reduced[:, 0], X_reduced[:, 1], X_reduced[:, 2], c=y)

cm = plt.cm.Set1

ax.set_title('First three PCA (Principal Component Analysis) directions')

ax.set_xlabel('1st eigenvector')

ax.waxis.set_ticklabels

ax.set_ylabel('2nd eigenvector')

ax.wyaxis.set_ticklabels

ax.set_zlabel('3rd eigenvector')

ax.wzaxis.set_ticklabels

plt.show()

Total running time of the script: 0 minutes 00.59 seconds

Note: Click [here](#) to download the full example code

5103 Plot randomly generated classification dataset

Plot several randomly generated 2D classification datasets. This example illustrates the datasets

make_classification, make_blobs, and datasets.make_gaussian_quantiles functions

For make_classification, three binary and two multiclass classification datasets are generated with different

numbers of informative features and clusters per class

510 Dataset examples 965

```
scikitlearn user guide Release 0213
printdoc
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.datasets import make_blobs
from sklearn.datasets import make_gaussian_quantiles
plt.figure(figsize=(8, 8))
plt.subplots_adjust(bottom=0.05, top=0.9, left=0.05, right=0.95)
plt.subplot(3, 2, 1)
plt.title('One informative feature, one cluster per class', fontsize=small)
X1, Y1 = make_classification(n_features=2, n_redundant=0, n_informative=1,
                             n_clusters_per_class=1)
966 Chapter 5 Examples
```

scikitlearn user guide Release 0213

pltscatterX1 0 X1 1 markero cY1

s25 edgecolork

pltsubplot322

plttitleTwo informative features one cluster per class fontsizesmall

X1 Y1 makeclassificationnfeatures2 nredundant0 ninformative2

nclustersperclass1

pltscatterX1 0 X1 1 markero cY1

s25 edgecolork

pltsubplot323

plttitleTwo informative features two clusters per class

fontsizesmall

X2 Y2 makeclassificationnfeatures2 nredundant0 ninformative2

pltscatterX2 0 X2 1 markero cY2

s25 edgecolork

pltsubplot324

plttitleMulticlass two informative features one cluster

fontsizesmall

X1 Y1 makeclassificationnfeatures2 nredundant0 ninformative2

nclustersperclass1 nclasses3

pltscatterX1 0 X1 1 markero cY1

s25 edgecolork

pltsubplot325

plttitleThree blobs fontsizesmall

X1 Y1 makeblobsnfeatures2 centers3

pltscatterX1 0 X1 1 markero cY1

s25 edgecolork

pltsubplot326

plttitleGaussian divided into three quantiles fontsizesmall

X1 Y1 makegaussianquantilesnfeatures2 nclasses3

pltscatterX1 0 X1 1 markero cY1

s25 edgecolork

pltshow

Total running time of the script 0 minutes 0097 seconds

Note Click here to download the full example code

5104 Plot randomly generated multilabel dataset

This illustrates the datasetsmakemultilabelclassification dataset generator Each sample consists of counts of two features up to 50 in total which are differently distributed in each of two classes Points are labeled as follows where Y means the class is present

510 Dataset examples 967

scikitlearn user guide Release 0213

123Color
Y N N Red
N Y N Blue
N N Y Yellow
Y Y N Purple
Y N Y Orange
Y Y N Green
Y Y Y Brown

A star marks the expected sample for each class its size reflects the probability of selecting that class label
The left and right examples highlight the nlabels parameter more of the samples in the right plot have 2 or 3 labels
Note that this twodimensional example is very degenerate generally the number of features would be much greater than the “document length” while here we have much larger documents than vocabulary Similarly with nclasses nfeatures it is much less likely that a feature distinguishes a particular class

Out
The data was generated from randomstate1013
Class PC Pw0C Pw1C
red 064 097 003
blue 006 060 040
yellow 030 009 091
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_multilabel_classification as make_mlclf
968 Chapter 5 Examples

scikitlearn user guide Release 0213

printdoc
COLORS nparray
FF3333 red
0198E1 blue
BF5FFF purple
FCD116 yellow
FF7216 orange
4DBD33 green
87421F brown

Use same random seed for multiple calls to makemultilabelclassification to ensure same distributions

RANDOMSEED np.random.randint(2, 10)
def plot2dax(nlabels=1, nclasses=3, length=50,
X=Y, pc=pwc, makemlclfnsamples=150, nfeatures=2,
nclasses=nclasses, nlabels=nlabels,
length=length, allowunlabeled=False,
returndistributions=True,
randomstate=RANDOMSEED,
axscatterX=0, X=1, color=COLOR, stakeY=1, 2, 4,
sumaxis=1,
marker=
axscatterpwc0=length, pwc1=length,
marker=, linewidth=5, edgecolor=black,
s20=1500, pc2=
color=COLOR, stake=1, 2, 4,
axsetxlabelFeature=0, count=
return pc, pwc=
ax1, ax2, plt.subplots(1, 2, sharex=True, sharey=True, figsize=(8, 4),
plt.subplots_adjust(bottom=15),
pc, pwc, plot2dax(1, nlabels=1,
ax1.set_title(nlabels=1, length=50,
ax1.set_xlabelFeature(1, count=
plot2dax(2, nlabels=3,
ax2.set_title(nlabels=3, length=50,
ax2.set_xlim(left=0, auto=True),
ax2.set_ylim(bottom=0, auto=True),
plt.show)
print('The data was generated from randomstate d ', RANDOMSEED)
print('Class PC Pw0C Pw1C sep t
fork p, pw, inzipred, blue, yellow, pc, pwcT
print(st02ft02ft02f k p pw0 pw1
Total running time of the script: 0 minutes 0085 seconds
510 Dataset examples 969

scikitlearn user guide Release 0213

511 Decomposition

Examples concerning the sklearndecomposition module

Note [Click here to download the full example code](#)

5111 Betadivergence loss functions

A plot that compares the various Betadivergence loss functions supported by the MultiplicativeUpdate ‘mu’ solver

insklearndecompositionNMF

import numpy as np

import matplotlib.pyplot as plt

from sklearndecompositionnmf import betadivergence

printdoc

x np.linspace(0,1, 4, 1000)

y np.zeros(x.shape)

colors mbg

forj beta in enumerate(0, 0.5, 1, 1.5, 2)

970 Chapter 5 Examples

scikitlearn user guide Release 0213

fori xiinenumeratex

yi betadivergence1 xi 1 beta

name beta 11f beta

pltplotx y labelname colorcolorsj

pltxlabelx

plttitlebetadivergence1 x

pltlegendloc0

pltaxis0 4 0 3

pltshow

Total running time of the script 0 minutes 0225 seconds

Note Click here to download the full example code

5112 PCA (Principal Component Analysis) example with Iris Dataset

Principal Component Analysis applied to the Iris dataset

See here for more information on this dataset

printdoc

Code source Gaël Varoquaux

License BSD (Binary Search Decision) 3 clause

import numpy as np

import matplotlib.pyplot as plt

from mpl_toolkits.mplot3d import Axes3D

from sklearn import decomposition

from sklearn import datasets

511 Decomposition 971

scikitlearn user guide Release 0213

nprandomseed5

centers 1 1 1 1 1 1

iris datasetsloadiris

X irisdata

y iristarget

fig pltfigure1 figsize4 3

pltclf

ax Axes3Dfig rect0 0 95 1 elev48 azim134

pltcla

pca decompositionPCAncomponents3

pcafitX

X pcatransformX

forname label inSetosa 0 Versicolour 1 Virginica 2

axtext3DXy label 0mean

Xy label 1mean 15

Xy label 2mean name

horizontalalignmentcenter

bboxdictalpha5 edgecolorw facecolorw

Reorder the labels to have colors matching the cluster results

y npchoosey 1 2 0astypepfloa

axscatterX 0 X 1 X 2 cy cmappltcmnipspectral

edgecolork

axwxaxissetticklabels

axwyaxissetticklabels

axwzaxissetticklabels

pltshow

Total running time of the script 0 minutes 0186 seconds

Note Click here to download the full example code

5113 Incremental PCA (Principal Component Analysis)

Incremental principal component analysis IPCA is typically used as a replacement for principal component analysis

PCA (Principal Component Analysis) when the dataset to be decomposed is too large to fit in memory IPCA builds a lowrank approximation for the input data using an amount of memory which is independent of the number of input data samples It is still dependent on the input data features but changing the batch size allows for control of memory usage

This example serves as a visual check that IPCA is able to find a similar projection of the data to PCA (Principal Component Analysis) to a sign flip while only processing a few samples at a time This can be considered a “toy example” as IPCA is intended for large datasets which do not fit in main memory requiring incremental approaches

972 Chapter 5 Examples

scikitlearn user guide Release 0213

- 511 Decomposition 973

scikitlearn user guide Release 0213

```
•
printdoc
Authors Kyle Kastner
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import loadiris
from sklearn.decomposition import PCA (Principal Component Analysis) IncrementalPCA
iris = loadiris
X = iris.data
y = iris.target
ncomponents = 2
ipca = IncrementalPCAncomponents=ncomponents batchsize=10
974 Chapter 5 Examples
```

```

scikitlearn user guide Release 0213
Xipca ipcafittransformX
pca PCAncomponentsncomponents
Xpca pcafittransformX
colors navy turquoise darkorange
forXtransformed title inXipca Incremental PCA (Principal Component Analysis) Xpca PCA (Principal Component Analysis)
pltfigurefigsize8 8
forcolor i targetname inzipcolors 0 1 2 iristargetnames
pltscatterXtransformedy i 0 Xtransformedy i 1
colorcolor lw2 labeltargetname
ifIncremental intitle
err npabsnpabsXpca npabsXipcamean
plttitletitle of iris dataset nMean absolute unsigned error
6f err
else
plttitletitle of iris dataset
pltlegendlocbest shadowFalse scatterpoints1
pltaxis4 4 15 15
pltshow
Total running time of the script 0 minutes 0143 seconds
Note Click here to download the full example code
5114 Comparison of LDA and PCA (Principal Component Analysis) 2D projection of Iris dataset
The Iris dataset represents 3 kind of Iris flowers Setosa Versicolour and Virginica with 4 attributes sepal length
sepal width petal length and petal width
Principal Component Analysis PCA (Principal Component Analysis) applied to this data identifies the combination of attributes principal compo
nents or directions in the feature space that account for the most variance in the data Here we plot the different
samples on the 2 first principal components
Linear Discriminant Analysis LDA tries to identify attributes that account for the most variance between classes In
particular LDA in contrast to PCA (Principal Component Analysis) is a supervised method using known class labels
511 Decomposition 975
```


scikitlearn user guide Release 0213

```
•
Out
explained variance ratio first two components 092461872 005306648
printdoc
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.discriminantanalysis import LinearDiscriminantAnalysis
iris = datasets.load_iris
X = iris.data
y = iris.target
target_names = iris.target_names
pca = PCA(n_components=2)
Xr = pca.fit_transform(X)
lda = LinearDiscriminantAnalysis(n_components=2)
511 Decomposition 977
```

```
scikitlearn user guide Release 0213
Xr2 IdafitX ytransformX
Percentage of variance explained for each components
printexplained variance ratio first two components s
strpcaexplainedvarianceratio
pltfigure
colors navy turquoise darkorange
lw 2
forcolor i targetname inzipcolors 0 1 2 targetnames
pltscatterXry i 0 Xry i 1 colorcolor alpha8 lwlw
labeltargetname
pltlegendlocbest shadowFalse scatterpoints1
plttitlePCA of IRIS dataset
pltfigure
forcolor i targetname inzipcolors 0 1 2 targetnames
pltscatterXr2y i 0 Xr2y i 1 alpha8 colorcolor
labeltargetname
pltlegendlocbest shadowFalse scatterpoints1
plttitleLDA of IRIS dataset
pltshow
Total running time of the script 0 minutes 0057 seconds
Note Click here to download the full example code
5115 Blind source separation using FastICA
An example of estimating sources from noisy data
Independent component analysis ICA is used to estimate sources given noisy measurements Imagine 3 instruments
playing simultaneously and 3 microphones recording the mixed signals ICA is used to recover the sources ie what is
played by each instrument Importantly PCA (Principal Component Analysis) fails at recovering our instruments since the related signals reflect
nonGaussian processes
978 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal
from sklearn.decomposition import FastICA PCA (Principal Component Analysis)
```

```
Generate sample data
nprandomseed0
nsamples 2000
time nplinspace0 8 nsamples
s1 npsin2 time Signal 1 sinusoidal signal
s2 npsignnpsin3 time Signal 2 square signal
s3 signalsawtooth2 nppitime Signal 3 saw tooth signal
S npcs1 s2 s3
S 02 nprandomnormalsizeSshape Add noise
S Sstdaxis0 Standardize data
Mix data
A nparray1 1 1 05 2 10 15 10 20 Mixing matrix
X npdotS AT Generate observations
511 Decomposition 979
```

```
scikitlearn user guide Release 0213
Compute ICA
ica FastICAncomponents3
S icafittransformX Reconstruct signals
A icamixing Get estimated mixing matrix
We can prove that the ICA model applies by reverting the unmixing
assertnpallcloseX npdotS AT icamean
For comparison compute PCA (Principal Component Analysis)
pca PCAncomponents3
H pcafittransformX Reconstruct signals based on orthogonal components
```

```
Plot results
pltfigure
models X S S H
names Observations mixed signal
True Sources
ICA recovered signals
PCA (Principal Component Analysis) recovered signals
colors red steelblue orange
forii model name inenumeratezipmodels names 1
pltsubplot4 1 ii
plttitlename
forsig color inzipmodelT colors
pltplotsig colorcolor
pltsubplotsadjust009 004 094 094 026 046
pltshow
Total running time of the script 0 minutes 0089 seconds
Note Click here to download the full example code
5116 Principal components analysis PCA (Principal Component Analysis)
These figures aid in illustrating how a point cloud can be very flat in one direction—which is where PCA (Principal Component Analysis) comes in to
choose a direction that is not flat
980 Chapter 5 Examples
```

scikitlearn user guide Release 0213

-
-

printdoc

Authors Gael Varoquaux

Jaques Grobler

Kevin Hughes

License BSD (Binary Search Decision) 3 clause

from sklearn.decomposition import PCA (Principal Component Analysis)

from mpl_toolkits.mplot3d import Axes3D

import numpy as np

import matplotlib.pyplot as plt

from scipy import stats

Create the data

511 Decomposition 981

scikitlearn user guide Release 0213

```
e npexp1
nprandomseed4
defpdfx
return05statsnormscale025 epdfx
statsnormscale4 epdfx
y nprandomnormalscale05 size30000
x nprandomnormalscale05 size30000
z nprandomnormalscale01 sizelenx
density pdfx pdfy
pdfz pdf5 z
density pdfz
a x y
b 2y
c a b z
norm npsqrtavar bvar
a norm
b norm
```

```
Plot the figures
defplotfigsfignum elev azim
fig pltfigurefignum figsize4 3
pltclf
ax Axes3Dfig rect0 0 95 1 elelev azimazim
axscattera10 b10 c10 cdensity10 marker alpha4
Y npca b c
Using SciPys SVD this would be
pcascore V scipylinalgsvdY fullmatricesFalse
pca PCAncomponents3
pcafitY
pcascore pcaexplainedvarianceratio
V pcacomponents
xpcaaxis ypcaaxis zpcaaxis 3 VT
xpcaplane nprxpcaaxis2 xpcaaxis11
ypcaplane nprypcaaxis2 ypcaaxis11
zpcaplane nprzpcaaxis2 zpcaaxis11
xpcaplaneshape 2 2
ypcaplaneshape 2 2
zpcaplaneshape 2 2
axplotsurfacexpcaplane ypcaplane zpcaplane
axwxaxissetticklabels
axwyaxissetticklabels
axwzaxissetticklabels
elev 40
982 Chapter 5 Examples
```

scikitlearn user guide Release 0213

```
azim 80
plotfigs1 elev azim
elev 30
azim 20
plotfigs2 elev azim
pltshow
```

Total running time of the script 0 minutes 0114 seconds

Note [Click here to download the full example code](#)

5117 FastICA on 2D point clouds

This example illustrates visually in the feature space a comparison by results using two different component analysis techniques

Independent component analysis ICA vsPrincipal component analysis PCA (Principal Component Analysis)

Representing ICA in the feature space gives the view of ‘geometric ICA’ ICA is an algorithm that finds directions in the feature space corresponding to projections with high nonGaussianity These directions need not be orthogonal in the original feature space but they are orthogonal in the whitened feature space in which all directions correspond to the same variance

PCA (Principal Component Analysis) on the other hand finds orthogonal directions in the raw feature space that correspond to directions accounting for maximum variance

Here we simulate independent sources using a highly nonGaussian process 2 student T with a low number of degrees

of freedom top left figure We mix them to create observations top right figure In this raw observation space

directions identified by PCA (Principal Component Analysis) are represented by orange vectors We represent the signal in the PCA (Principal Component Analysis)

whitening by the variance corresponding to the PCA (Principal Component Analysis) vectors lower left Running ICA corresponds to finding a

rotation in this space to identify the directions of largest nonGaussianity lower right

511 Decomposition 983

```
scikitlearn user guide Release 0213
printdoc
  Authors Alexandre Gramfort Gael Varoquaux
  License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA (Principal Component Analysis) FastICA

Generate sample data
rng = np.random.RandomState(42)
S = rng.standard_t(15, size=(20000, 2))
S = S * 2
Mix data
A = np.array([1, 0, 2]) Mixing matrix
X = np.dot(S, A.T) Generate observations
pca = PCA (Principal Component Analysis)
Spca = pca.fit_transform(X)
ica = FastICA(random_state=rng)
Sica = ica.fit_transform(X) Estimate the sources
984 Chapter 5 Examples
```

scikitlearn user guide Release 0213
Sica SicaStdaxis0

Plot results
defplotsamplesS axislistNone
pltscatterS 0 S 1 s2 markero zorder10
colorsteelblue alpha05
ifaxislist is notNone
colors orange red
forcolor axis inzipcolors axislist
axis axisstd
xaxis yaxis axis
Trick to get legend to work
pltplot01 xaxis 01 yaxis linewidth2 colorcolor
pltquiver0 0 xaxis yaxis zorder11 width001 scale6
colorcolor
plthlines0 3 3
pltvlines0 3 3
pltxlim3 3
pltylim3 3
pltlabelx
pltlabely
pltfigure
pltsubplot2 2 1
plotsamplesS Sstd
plttitleTrue Independent Sources
axislist pcacomponentsT icamixing
pltsubplot2 2 2
plotsamplesX npstdX axislistaxislist
legend pltlegendPCA ICA locupper right
legendsetzorder100
plttitleObservations
pltsubplot2 2 3
plotsamplesSpcap npstdSpcap axis0
plttitlePCA recovered signals
pltsubplot2 2 4
plotsamplesSica npstdSica
plttitleICA recovered signals
pltsubplotsadjust009 004 094 094 026 036
pltshow
Total running time of the script 0 minutes 0357 seconds
Note [Click here to download the full example code](#)
511 Decomposition 985

```
scikitlearn user guide Release 0213
5118 Kernel PCA (Principal Component Analysis)
This example shows that Kernel PCA (Principal Component Analysis) is able to find a projection of the data that makes data linearly separable
printdoc
Authors Mathieu Blondel
Andreas Mueller
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA (Principal Component Analysis) KernelPCA
from sklearn.datasets import make_circles
np.random.seed(0)
X, y = make_circles(n_samples=400, factor=3, noise=0.05)
kpca = KernelPCA(kernel='rbf', fit_inverse_transform=True, gamma=10)
Xkpca = kpca.fit_transform(X)
Xback = kpca.inverse_transform(Xkpca)
pca = PCA (Principal Component Analysis)
Xpca = pca.fit_transform(X)
986 Chapter 5 Examples
```


scikitlearn user guide Release 0213

```
Plot results
pltfigure
pltsubplot2 2 1 aspectequal
plttitleOriginal space
reds y 0
blues y 1
pltscatterXreds 0 Xreds 1 cred
s20 edgecolork
pltscatterXblues 0 Xblues 1 cblue
s20 edgecolork
pltlabelx1
pltlabelx2
X1 X2 npmeshgridnplinspace15 15 50 nplinspace15 15 50
Xgrid nparraynpravelX1 npravelX2T
projection on the first principal component in the phi space
Zgrid kpcatransformXgrid 0reshapeX1shape
pltcontourX1 X2 Zgrid colorsgrey linewidths1 originlower
pltsubplot2 2 2 aspectequal
pltscatterXpcareds 0 Xpcareds 1 cred
s20 edgecolork
pltscatterXpcablues 0 Xpcablues 1 cblue
s20 edgecolork
plttitleProjection by PCA (Principal Component Analysis)
pltlabel1st principal component
pltlabel2nd component
pltsubplot2 2 3 aspectequal
pltscatterXkpcareds 0 Xkpcareds 1 cred
s20 edgecolork
pltscatterXkpcablues 0 Xkpcablues 1 cblue
s20 edgecolork
plttitleProjection by KPCA
pltlabelr1st principal component in space induced by phi
pltlabel2nd component
pltsubplot2 2 4 aspectequal
pltscatterXbackreds 0 Xbackreds 1 cred
s20 edgecolork
pltscatterXbackblues 0 Xbackblues 1 cblue
s20 edgecolork
plttitleOriginal space after inverse transform
pltlabelx1
pltlabelx2
plttightlayout
pltshow
Total running time of the script 0 minutes 0460 seconds
Note Click here to download the full example code
511 Decomposition 987
```

scikitlearn user guide Release 0213

5119 Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA

Probabilistic PCA (Principal Component Analysis) and Factor Analysis are probabilistic models The consequence is that the likelihood of new data can be used for model selection and covariance estimation Here we compare PCA (Principal Component Analysis) and FA with crossvalidation on low rank data corrupted with homoscedastic noise noise variance is the same for each feature or heteroscedastic noise noise variance is the different for each feature In a second step we compare the model likelihood to the likelihoods obtained from shrinkage covariance estimators

One can observe that with homoscedastic noise both FA and PCA (Principal Component Analysis) succeed in recovering the size of the low rank subspace The likelihood with PCA (Principal Component Analysis) is higher than FA in this case However PCA (Principal Component Analysis) fails a when heteroscedastic noise is present Under appropriate circumstances the low rank models are more likely than shrinkage models

The automatic estimation from Automatic Choice of Dimensionality for PCA (Principal Component Analysis) NIPS 2000 598604 by Thomas P Minka is also compared

-

988 Chapter 5 Examples

scikitlearn user guide Release 0213

•

Out

best ncomponents by PCA (Principal Component Analysis) CV (Cross-Validation) 10

best ncomponents by FactorAnalysis CV (Cross-Validation) 10

best ncomponents by PCA (Principal Component Analysis) MLE 10

best ncomponents by PCA (Principal Component Analysis) CV (Cross-Validation) 35

best ncomponents by FactorAnalysis CV (Cross-Validation) 10

best ncomponents by PCA (Principal Component Analysis) MLE 38

Authors Alexandre Gramfort

Denis A Engemann

License BSD (Binary Search Decision) 3 clause

import numpy as np

import matplotlib.pyplot as plt

from scipy import linalg

from sklearn.decomposition import PCA (Principal Component Analysis) FactorAnalysis

from sklearn.covariance import ShrunkCovariance LedoitWolf

from sklearn.model_selection import cross_val_score

from sklearn.model_selection import GridSearchCV

511 Decomposition 989

scikitlearn user guide Release 0213
printdoc

Create the data
nsamples nfeatures rank 1000 50 10
sigma 1
rng np.random.RandomState(42)
U linalg.svd(rng.randn(nfeatures, nfeatures)
X np.dot(rng.randn(nsamples, rank), U) * rankT
Adding homoscedastic noise
Xhomo X * sigma * rng.randn(nsamples, nfeatures)
Adding heteroscedastic noise
sigmas sigma * rng.randn(nfeatures, sigma) * 2
Xhetero X * rng.randn(nsamples, nfeatures) * sigmas

Fit the models
ncomponents np.arange(0, nfeatures, 5) # options for ncomponents
def compute_scores(X):
 pca = PCA(svd_solver='full')
 fa = FactorAnalysis()
 pcascores, fascalcores =
 for n in ncomponents:
 pcascores.append(np.mean(cross_val_score(pca, X, cv=5)
 fascalcores.append(np.mean(cross_val_score(fa, X, cv=5)
 return pcascores, fascalcores
def shrink_cov_score(X):
 shrinkages = np.logspace(2, 0, 30)
 cv = GridSearchCV(ShrinkCovariance, shrinkage=shrinkages, cv=5)
 return np.mean(cross_val_score(cv.fit(X), X, cv=5)
 def lw_score(X):
 return np.mean(cross_val_score(LedoitWolf, X, cv=5)
 for X, title in [(Xhomo, 'Homoscedastic Noise'),
 (Xhetero, 'Heteroscedastic Noise')]:
 pcascores, fascalcores = compute_scores(X)
 ncomponents_pca, ncomponents_npargmax_pcascores =
 ncomponents_fa, ncomponents_npargmax_fascalcores =
 pca = PCA(svd_solver='full')
 pca.fit(X)
 ncomponents_pcamle, pcamle_components =
990 Chapter 5 Examples

```
scikitlearn user guide Release 0213
printbest ncomponents by PCA (Principal Component Analysis) CV (Cross-Validation) d ncomponentspca
printbest ncomponents by FactorAnalysis CV (Cross-Validation) d ncomponentsfa
printbest ncomponents by PCA (Principal Component Analysis) MLE d ncomponentspcamle
pltfigure
pltplotncomponents pcascores b labelPCA scores
pltplotncomponents fascores r labelFA scores
pltaxvlinerrank colorg labelTRUTH d rank linestyle
pltaxvlinencomponentspca colorb
labelPCA CV (Cross-Validation) d ncomponentspca linestyle
pltaxvlinencomponentsfa colorr
labelFactorAnalysis CV (Cross-Validation) d ncomponentsfa
linestyle
pltaxvlinencomponentspcamle colork
labelPCA MLE d ncomponentspcamle linestyle
compare with other covariance estimators
pltaxhlineshrunkcovscoreX colorviolet
labelShrunk Covariance MLE linestyle
pltaxhlinelwscoreX colororange
labelLedoitWolf MLE ncomponentspcamle linestyle
pltxlabelnb of components
pltylabelCV scores
pltlegendloclower right
plttitletitle
pltshow
Total running time of the script 0 minutes 17528 seconds
Note Click here to download the full example code
51110 Sparse coding with a precomputed dictionary
Transform a signal as a sparse combination of Ricker wavelets This example visually compares different sparse coding
methods using the sklearndecompositionSparseCoder estimator The Ricker also known as Mexican
hat or the second derivative of a Gaussian is not a particularly good kernel to represent piecewise constant signals
like this one It can therefore be seen how much adding different widths of atoms matters and it therefore motivates
learning the dictionary to best fit your type of signals
The richer dictionary on the right is not larger in size heavier subsampling is performed in order to stay on the same
order of magnitude
511 Decomposition 991
```

```
scikitlearn user guide Release 0213
printdoc
from distutilsversion import LooseVersion
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import SparseCoder
def rickerfunction(resolution, center, width):
    Discrete subsampled Ricker Mexican hat wavelet
    x = np.linspace(0, resolution, 1, resolution)
    x = 2 * np.sqrt(3) * width * np.pi * 1 / 4
    1 * x - center * 2 * width * 2
    np.exp(x - center * 2 * 2 * width * 2)
    return x
def rickermatrix(width, resolution, ncomponents):
    Dictionary of Ricker Mexican hat wavelets
    centers = np.linspace(0, resolution, 1, ncomponents)
    D = np.empty(ncomponents, resolution)
    for i, center in enumerate(centers):
        Di = rickerfunction(resolution, center, width)
    D = np.sqrt(np.sum(D ** 2, axis=1)) * np.newaxis
    return D
resolution = 1024
subsampling = 3 # subsampling factor
width = 100
ncomponents = resolution // subsampling
# Compute a wavelet dictionary
Dfixed = rickermatrix(width, width, resolution)
ncomponents = ncomponents
Dmulti = np.r_[Dfixed, rickermatrix(width, width, resolution)]
992 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
ncomponentsncomponents 5
forwin10 50 100 500 1000
Generate a signal
y nplinspace0 resolution 1 resolution
firstquarter y resolution 4
yfirstquarter 3
ynplogicalnotfirstquarter 1
List the different sparse coding methods in the following format
title transformalgorithm transformalpha
transformnnozero coeffs color
estimators OMP omp None 15 navy
Lasso lassolars 2 None turquoise
lw 2
Avoid FutureWarning about default value change when numpy 114
lstsqrcond None if LooseVersion npversion 114 else 1
pltfigurefigsize13 6
forsubplot D title in enumerate zip D fixed D multi
fixed width multiple widths
pltsubplot1 2 subplot 1
plttitle Sparse coding against sdictionary title
pltploty lwlw linestyle label Original signal
Do a wavelet approximation
fortitle algo alpha nnonzero color in estimators
coder SparseCoder dictionary D transform nnonzero coeffs nnonzero
transform alpha alpha transform algo
x coder transform y reshape 1 1
density len np flat nnonzero x
x npravel np dot x D
squarederror np sum y x 2
pltplotx color color lwlw
label ssnonzero coeffs n 2 ferror
title density squarederror
Soft thresholding debiasing
coder SparseCoder dictionary D transform algo r m threshold
transform alpha 20
x coder transform y reshape 1 1
idx np where x 0
x0 idx nplinalg lstsq D idx T y rcond lstsq rcond
x npravel np dot x D
squarederror np sum y x 2
pltplotx color darkorange lwlw
label Thresholding w debiasing ndnonzero coeffs 2 ferror
len idx squarederror
pltaxis tight
pltlegend shadow False loc best
pltsubplots adjust 0 4 0 7 9 7 9 0 9 2
pltshow
Total running time of the script 0 minutes 0238 seconds
Note Click here to download the full example code
511 Decomposition 993
```

scikitlearn user guide Release 0213

51111 Image denoising using dictionary learning

An example comparing the effect of reconstructing noisy fragments of a raccoon face image using firstly online Dictionary Learning and various transform methods

The dictionary is fitted on the distorted left half of the image and subsequently used to reconstruct the right half Note that even better performance could be achieved by fitting to an undistorted ie noiseless image but here we start from the assumption that it is not available

A common practice for evaluating the results of image denoising is by looking at the difference between the reconstruction and the original image If the reconstruction is perfect this will look like Gaussian noise

It can be seen from the plots that the results of Orthogonal Matching Pursuit OMP with two nonzero coefficients is a bit less biased than when keeping only one the edges look less prominent It is in addition closer from the ground truth in Frobenius norm

The result of Least Angle Regression is much more strongly biased the difference is reminiscent of the local intensity value of the original image

Thresholding is clearly not useful for denoising but it is here to show that it can produce a suggestive output with very high speed and thus be useful for other tasks such as object classification where performance is not necessarily related to visualisation

•

scikitlearn user guide Release 0213

-
-

511 Decomposition 995

-
-

scikitlearn user guide Release 0213

•

Out

Distorting image

Extracting reference patches

done in 001s

Learning the dictionary

done in 722s

Extracting noisy patches

done in 000s

Orthogonal Matching Pursuit

1 atom

done in 119s

Orthogonal Matching Pursuit

2 atoms

done in 235s

Leastangle regression

5 atoms

done in 1750s

Thresholding

alpha01

done in 025s

printdoc

from time import time

import matplotlib.pyplot as plt

import numpy as np

import scipy as sp

from sklearn.decomposition import MiniBatchDictionaryLearning

511 Decomposition 997

```
scikitlearn user guide Release 0213
from sklearnfeatureextractionimage import extractpatches2d
from sklearnfeatureextractionimage import reconstructfrompatches2d
try SciPy 016 have face in misc
from scipymisc import face
face facegrayTrue
exceptImportError
face spfacegrayTrue
Convert from uint8 representation with values between 0 and 255 to
a floating point representation with values between 0 and 1
face face 255
downsample for higher speed
face face4 4 face14 4 face4 14 face14 14
face 40
height width faceshape
Distort the right half of the image
printDistorting image
distorted facecopy
distorted width 2 0075 nprandomrandnheight width 2
Extract all reference patches from the left half of the image
printExtracting reference patches
t0 time
patchsize 7 7
data extractpatches2ddistorted width 2 patchsize
data datareshapedatashape0 1
data npmeandata axis0
data npstddata axis0
printdone in 2fs time t0

Learn the dictionary from reference patches
printLearning the dictionary
t0 time
dico MiniBatchDictionaryLearningncomponents100 alpha1 niter500
V dicofitdatacomponents
dt time t0
printdone in 2fs dt
pltfigurefigsize42 4
fori comp inenumerateV100
pltsubplot10 10 i 1
pltimshowcompreshapepatchsize cmappltcmgrayr
interpolationnearest
pltxticks
pltyticks
pltsuptitleDictionary learned from face patches n
Train time 1fs ondpatches dt lendata
fontsize16
pltsubplotsadjust008 002 092 085 008 023
```

```
scikitlearn user guide Release 0213
    Display the distorted image
defshowwithdiffimage reference title
Helper function to display denoising
pltfigurefigsize5 33
pltsubplot1 2 1
plttitleImage
pltimshowimage vmin0 vmax1 cmappltcmgray
interpolationnearest
pltxticks
pltyticks
pltsubplot1 2 2
difference image reference
plttitleDifference norm 2f npsqrtnpsumdifference 2
pltimshowdifference vmin05 vmax05 cmappltcmPuOr
interpolationnearest
pltxticks
pltyticks
pltstitletitle size16
pltsubplotsadjust002 002 098 079 002 02
showwithdiffdistorted face Distorted image
```

```
    Extract noisy patches and reconstruct them using the dictionary
printExtracting noisy patches
t0 time
data extractpatches2ddistorted width 2 patchsize
data datareshapedatashape0 1
intercept npmeandata axis0
data intercept
printdone in 2fs time t0
transformalgorithms
Orthogonal Matching Pursuit n1 atom omp
transformnnonzerocoefs 1
Orthogonal Matching Pursuit n2 atoms omp
transformnnonzerocoefs 2
Leastangle regression n5 atoms lars
transformnnonzerocoefs 5
Thresholding nalpha01 threshold transformalpha 1
reconstructions
fortitle transformalgorithm kwargs intransformalgorithms
printtitle
reconstructionstitle facecopy
t0 time
dicosetparamstrtransformalgorithmtransformalgorithm kwargs
code dicotransformdata
patches npdotcode V
patches intercept
patches patchesreshapelendata patchsize
iftransformalgorithm threshold
patches patchesmin
patches patchesmax
511 Decomposition 999
```

```
scikitlearn user guide Release 0213
reconstructionstitle width 2 reconstructfrompatches2d
patches height width 2
dt time t0
printdone in 2fs dt
showwithdiffreconstructionstitle face
title time 1fs dt
pltshow
Total running time of the script 0 minutes 30244 seconds
Note Click here to download the full example code
51112 Faces dataset decompositions
This example applies to olivettifaces different unsupervised matrix decomposition dimension reduction methods
from the module sklearndecomposition see the documentation chapter Decomposing signals in components
matrix factorization problems
•
1000 Chapter 5 Examples
```

scikitlearn user guide Release 0213

- 511 Decomposition 1001

scikitlearn user guide Release 0213

- 511 Decomposition 1003

scikitlearn user guide Release 0213

- 511 Decomposition 1005

scikitlearn user guide Release 0213

-
-

1006 Chapter 5 Examples

scikitlearn user guide Release 0213

-

511 Decomposition 1007

scikitlearn user guide Release 0213

•

511 Decomposition 1009

scikitlearn user guide Release 0213

- 511 Decomposition 1011

scikitlearn user guide Release 0213

•

Out

Dataset consists of 400 faces

Extracting the top 6 Eigenfaces PCA (Principal Component Analysis) using randomized SVD
done in 0018s

Extracting the top 6 Nonnegative components NMF (Non-Negative Matrix Factorization)
done in 0109s

Extracting the top 6 Independent components FastICA
done in 0295s

Extracting the top 6 Sparse comp MiniBatchSparsePCA
done in 1129s

Extracting the top 6 MiniBatchDictionaryLearning
done in 0979s

Extracting the top 6 Cluster centers MiniBatchKMeans
done in 0221s

Extracting the top 6 Factor Analysis components FA
done in 0206s

Extracting the top 6 Dictionary learning
done in 1099s

Extracting the top 6 Dictionary learning positive dictionary
done in 1213s

Extracting the top 6 Dictionary learning positive code
done in 0688s

Extracting the top 6 Dictionary learning positive dictionary code
done in 0470s

1012 Chapter 5 Examples

scikitlearn user guide Release 0213

printdoc

Authors Vlad Niculae Alexandre Gramfort

License BSD (Binary Search Decision) 3 clause

import logging

from time import time

from numpyrandom import RandomState

import matplotlib.pyplot as plt

from sklearn.datasets import fetcholivettifaces

from sklearn.cluster import MiniBatchKMeans

from sklearn import decomposition

Display progress logs on stdout

logging.basicConfig(level=logging.INFO)

format asctime level names messages

nrow ncol 2 3

ncomponents nrow ncol

imageshape 64 64

rng RandomState(0)

Load faces data

dataset fetcholivettifaces shuffle=True randomstate=rng

faces dataset.data

nsamples nfeatures faceshape

global centering

facescentered faces faces.mean(axis=0)

local centering

facescentered facescentered.mean(axis=1).reshape(samples, 1)

printDataset consists of dfaces nsamples

def plot_gallery(title, images, ncol, nrow, cmap=plt.cm.gray)

plt.figure(figsize=(2, ncol), dpi=226, nrow=nrow)

plt.suptitle(title, size=16)

for i, comp in enumerate(images):

plt.subplot(nrow, ncol, i + 1)

vmax = max(comp.max(), comp.min())

plt.imshow(comp.reshape(imageshape), cmap=cm)

interpolation=nearest)

vmin, vmax = comp.min(), comp.max()

plt.xticks(

plt.yticks(

plt.subplots_adjust(0.01, 0.05, 0.99, 0.93, 0.04, 0)

List of the different estimators whether to center and transpose the

problem and whether the transformer uses the clustering API (Application Programming Interface)

estimators

Eigenfaces PCA (Principal Component Analysis) using randomized SVD

decomposition PCA ncomponents ncomponents svd solver randomized

511 Decomposition 1013

```
scikitlearn user guide Release 0213
whitenTrue
True
Nonnegative components NMF (Non-Negative Matrix Factorization)
decompositionNMFncomponentsncomponents initnndsvda tol5e3
False
Independent components FastICA
decompositionFastICAncomponentsncomponents whitenTrue
True
Sparse comp MiniBatchSparsePCA
decompositionMiniBatchSparsePCAncomponentsncomponents alpha08
niter100 batchsize3
randomstaterng
normalizecomponentsTrue
True
MiniBatchDictionaryLearning
decompositionMiniBatchDictionaryLearningncomponents15 alpha01
niter50 batchsize3
randomstaterng
True
Cluster centers MiniBatchKMeans
MiniBatchKMeansnclustersncomponents tol1e3 batchsize20
maxiter50 randomstaterng
True
Factor Analysis components FA
decompositionFactorAnalysisncomponentsncomponents maxiter20
True
```

```
Plot a sample of the input data
plotgalleryFirst centered Olivetti faces facescenteredncomponents
```

```
Do the estimation and plot it
forname estimator center inestimators
printExtracting the top d s ncomponents name
t0 time
data faces
ifcenter
data facescentered
estimatorfitdata
traintime time t0
printdone in 03fs traintime
ifhasattrestimator clustercenters
components estimatorclustercenters
else
components estimatorcomponents
Plot an image representing the pixelwise variance provided by the
1014 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
estimator eg its noisevariance attribute The Eigenfaces estimator
via the PCA (Principal Component Analysis) decomposition also provides a scalar noisevariance
the mean of pixelwise variance that cannot be displayed as an image
so we skip it
ifhasattrestimator noisevariance and
estimatornoisevariancendim 0 Skip the Eigenfaces case
plotgalleryPixelwise variance
estimatornoisevariancereshape1 1 ncol1
nrow1
plotgallery s Train time 1fs name traintime
componentsncomponents
pltshow
```

```
Various positivity constraints applied to dictionary learning
estimators
Dictionary learning
decompositionMiniBatchDictionaryLearningncomponents15 alpha01
niter50 batchsize3
randomstaterng
True
Dictionary learning positive dictionary
decompositionMiniBatchDictionaryLearningncomponents15 alpha01
niter50 batchsize3
randomstaterng
positivedictTrue
True
Dictionary learning positive code
decompositionMiniBatchDictionaryLearningncomponents15 alpha01
niter50 batchsize3
randomstaterng
positivecodeTrue
True
Dictionary learning positive dictionary code
decompositionMiniBatchDictionaryLearningncomponents15 alpha01
niter50 batchsize3
randomstaterng
positivedictTrue
positivecodeTrue
True
```

```
Plot a sample of the input data
plotgalleryFirst centered Olivetti faces facescenteredncomponents
cmapppltcmRdBu
```

```
Do the estimation and plot it
forname estimator center inestimators
printExtracting the top d s ncomponents name
t0 time
data faces
ifcenter
511 Decomposition 1015
```

scikitlearn user guide Release 0213
data facescentered
estimatorfitdata
traintime time t0
printdone in 03fs traintime
components estimatorcomponents
plotgalleryname componentsncomponents cmappltcmRdBu
pltshow
Total running time of the script 0 minutes 8614 seconds
512 Ensemble methods
Examples concerning the sklearnensemble module
Note Click here to download the full example code
5121 Decision Tree Regression with AdaBoost
A decision tree is boosted using the AdaBoostR21algorithm on a 1D sinusoidal dataset with a small amount of
Gaussian noise 299 boosts 300 decision trees is compared with a single decision tree regressor As the number of
boosts is increased the regressor can fit more detail
1
8 Drucker “Improving Regressors using Boosting Techniques” 1997
1016 Chapter 5 Examples

scikitlearn user guide Release 0213
printdoc
Author Noel Dawe noeldawegmailcom

License BSD (Binary Search Decision) 3 clause
importing necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import AdaBoostRegressor
Create the dataset
rng = np.random.RandomState(1)
X = np.linspace(0, 6, 100).reshape(100, 1)
y = np.sin(X) + rng.normal(0, 0.1, X.shape[0])
Fit regression model
regr1 = DecisionTreeRegressor(max_depth=4)
regr2 = AdaBoostRegressor(DecisionTreeRegressor(max_depth=4),
n_estimators=300, random_state=rng)
regr1.fit(X, y)
regr2.fit(X, y)
512 Ensemble methods 1017

scikitlearn user guide Release 0213

```
Predict
y1 = regr1.predict(X)
y2 = regr2.predict(X)
# Plot the results
plt.figure()
plt.scatter(X, y, c=ck_label, training_samples)
plt.plot(X, y1, c=cg_label, linewidth=2)
plt.plot(X, y2, c=cr_label, linewidth=2)
plt.xlabel(data)
plt.ylabel(target)
plt.title(Boosted Decision Tree Regression)
plt.legend()
plt.show()
```

Total running time of the script: 0 minutes 02.26 seconds

Note: [Click here to download the full example code](#)

5122 Pixel importances with a parallel forest of trees

This example shows the use of forests of trees to evaluate the importance of the pixels in an image classification task: faces. The hotter the pixel, the more important.

The code below also illustrates how the construction and the computation of the predictions can be parallelized within multiple jobs.

scikitlearn user guide Release 0213

Out
Fitting ExtraTreesClassifier on faces data with 1 cores
done in 1028s

```
printdoc
from time import time
import matplotlib.pyplot as plt
from sklearn.datasets import fetcholivettifaces
from sklearn.ensemble import ExtraTreesClassifier
    Number of cores to use to perform parallel fitting of the forest model
njobs 1
    Load the faces dataset
data fetcholivettifaces
X dataimagesreshapelendataimages 1
y datatarget
512 Ensemble methods 1019
```

```
scikitlearn user guide Release 0213
mask y 5 Limit to 5 classes
X Xmask
y ymask
Build a forest and compute the pixel importances
printFitting ExtraTreesClassifier on faces data with dcores njobs
t0 time
forest ExtraTreesClassifiernestimators1000
maxfeatures128
njobsnjobs
randomstate0
forestfitX y
printdone in 03fs time t0
importances forestfeatureimportances
importances importancesreshapedataimages0shape
Plot pixel importances
pltmatshowimportances cmappltcmhot
plttitlePixel importances with forests of trees
pltshow
Total running time of the script 0 minutes 1147 seconds
Note Click here to download the full example code
5123 Plot individual and voting regression predictions
Plot individual and averaged regression predictions for Boston dataset
First three exemplary regressors are initialized GradientBoostingRegressor
RandomForestRegressor andLinearRegression and used to initialize a VotingRegressor
The red starred dots are the averaged predictions
1020 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdoc
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearnensemble import GradientBoostingRegressor
from sklearnensemble import RandomForestRegressor
from sklearnlinear import LinearRegression
from sklearnensemble import VotingRegressor

Loading some example data
boston = datasets.load_boston()
X = boston.data
y = boston.target

Training classifiers
reg1 = GradientBoostingRegressor(random_state=1, n_estimators=10)
reg2 = RandomForestRegressor(random_state=1, n_estimators=10)
reg3 = LinearRegression()
ereg = VotingRegressor([reg1, reg2, reg3])

reg1.fit(X, y)
reg2.fit(X, y)
reg3.fit(X, y)
ereg.fit(X, y)

xt = X[20]
512 Ensemble methods 1021
```

scikitlearn user guide Release 0213

```
pltfigure
pltplotreg1predictxt gd labelGradientBoostingRegressor
pltplotreg2predictxt b labelRandomForestRegressor
pltplotreg3predictxt ys labelLinearRegression
pltploteregpredictxt r labelVotingRegressor
plttickparamsaxisx whichboth bottomFalse topFalse
labelbottomFalse
pltylabelpredicted
pltxlabeltraining samples
pltlegendlocbest
plttitleComparison of individual predictions with averaged
pltshow
```

Total running time of the script 0 minutes 0117 seconds

Note [Click here to download the full example code](#)

5124 Feature importances with forests of trees

This examples shows the use of forests of trees to evaluate the importance of features on an artificial classification task The red bars are the feature importances of the forest along with their intertrees variability

As expected the plot suggests that 3 features are informative while the remaining are not

1022 Chapter 5 Examples

scikitlearn user guide Release 0213

Out

Feature ranking

1 feature 1 0295902

2 feature 2 0208351

3 feature 0 0177632

4 feature 3 0047121

5 feature 6 0046303

6 feature 8 0046013

7 feature 7 0045575

8 feature 4 0044614

9 feature 9 0044577

10 feature 5 0043912

printdoc

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import make_classification

from sklearn.ensemble import ExtraTreesClassifier

512 Ensemble methods 1023

scikitlearn user guide Release 0213

Build a classification task using 3 informative features

X y makeclassificationnsamples1000

nfeatures10

ninformative3

nredundant0

nrepeated0

nclasses2

randomstate0

shuffleFalse

Build a forest and compute the feature importances

forest ExtraTreesClassifiernestimators250

randomstate0

forestfitX y

importances forestfeatureimportances

std npstdtreefeatureimportances fortreeinforestestimators

axis0

indices npargsortimportances1

Print the feature ranking

printFeature ranking

forfinrangeXshape1

printd feature df f 1 indicesf importancesindicesf

Plot the feature importances of the forest

pltfigure

plttitleFeature importances

pltbarrangeXshape1 importancesindices

colorr yerrstdindices aligncenter

pltxticksrangeXshape1 indices

pltxlim1 Xshape1

pltshow

Total running time of the script 0 minutes 0343 seconds

Note Click here to download the full example code

5125 IsolationForest example

An example using sklearnensembleIsolationForest for anomaly detection

The IsolationForest ‘isolates’ observations by randomly selecting a feature and then randomly selecting a split value

between the maximum and minimum values of the selected feature

Since recursive partitioning can be represented by a tree structure the number of splittings required to isolate a sample

is equivalent to the path length from the root node to the terminating node

This path length averaged over a forest of such random trees is a measure of normality and our decision function

Random partitioning produces noticeable shorter paths for anomalies Hence when a forest of random trees collec

tively produce shorter path lengths for particular samples they are highly likely to be anomalies

1024 Chapter 5 Examples

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearnensemble import IsolationForest
rng = np.random.RandomState(42)
Generate train data
X = 0.3 * rng.randn(100, 2)
Xtrain = np.concatenate((X, X), axis=0)
Generate some regular novel observations
X = 0.3 * rng.randn(20, 2)
Xtest = np.concatenate((X, X), axis=0)
Generate some abnormal novel observations
Xoutliers = rng.uniform(low=-4, high=4, size=(20, 2))
fit the model
clf = IsolationForestbehaviour=new, max_samples=100,
random_state=rng, contamination='auto')
clf.fit(Xtrain)
ypredtrain = clf.predict(Xtrain)
ypredtest = clf.predict(Xtest)
ypredoutliers = clf.predict(Xoutliers)
plot the line the samples and the nearest vectors to the plane
```

scikitlearn user guide Release 0213
xx yy npmeshgridnplinspace5 5 50 nplinspace5 5 50
Z clfdecisionfunctionnpcxxravel yyravel
Z Zreshapexxshape
plttitleIsolationForest
pltcontourfxx yy Z cmappltcmBluesr
b1 pltscatterXtrain 0 Xtrain 1 cwhite
s20 edgecolork
b2 pltscatterXtest 0 Xtest 1 cgreen
s20 edgecolork
c pltscatterXoutliers 0 Xoutliers 1 cred
s20 edgecolork
pltaxistight
pltxlim5 5
pltylim5 5
pltlegendb1 b2 c
training observations
new regular observations new abnormal observations
locupper left
pltshow
Total running time of the script 0 minutes 0237 seconds
Note Click here to download the full example code
5126 Plot the decision boundaries of a VotingClassifier
Plot the decision boundaries of a VotingClassifier for two features of the Iris dataset
Plot the class probabilities of the first sample in a toy dataset predicted by three different classifiers and averaged by
theVotingClassifier
First three exemplary classifiers are initialized DecisionTreeClassifier KNeighborsClassifier and
SVC (Support Vector Classification) and used to initialize a softvoting VotingClassifier with weights 2 1 2 which means that the
predicted probabilities of the DecisionTreeClassifier andSVC count 5 times as much as the weights of the
KNeighborsClassifier classifier when the averaged probability is calculated
1026 Chapter 5 Examples


```
scikitlearn user guide Release 0213
printdoc
from itertools import product
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC (Support Vector Classification)
from sklearn.ensemble import VotingClassifier
Loading some example data
iris = datasets.load_iris()
X = iris.data[:, 2:]
y = iris.target
Training classifiers
clf1 = DecisionTreeClassifier(max_depth=4)
clf2 = KNeighborsClassifier(n_neighbors=7)
clf3 = SVC(gamma=1, kernel='rbf', probability=True)
ecf = VotingClassifier(estimators=[('dt', clf1), ('knn', clf2), ('svm', clf3)])
512 Ensemble methods 1027
```

scikitlearn user guide Release 0213

```
svc clf3
votingsoft weights2 1 2
clf1fitX y
clf2fitX y
clf3fitX y
ecclf fitX y
    Plotting decision regions
xmin xmax X 0min 1 X 0max 1
ymin ymax X 1min 1 X 1max 1
xx yy npmeshgridnparangexmin xmax 01
nparangeymaxmin ymax 01
f axarr pltsubplots2 2 sharexcol shareyrow figsize10 8
foridx clf tt inzipproduct0 1 0 1
clf1 clf2 clf3 eclf
Decision Tree depth4 KNN k7
Kernel SVM (Support Vector Machine) Soft Voting
Z clfpredictnpcxxravel yy.ravel
Z Zreshapexxshape
axarridx0 idx1contourfxx yy Z alpha04
axarridx0 idx1scatterX 0 X 1 cy
s20 edgecolork
axarridx0 idx1settitlett
pltshow
```

Total running time of the script 0 minutes 0202 seconds

Note [Click here to download the full example code](#)

5127 Comparing random forests and the multioutput meta estimator

An example to compare multioutput regression with random forest and the multioutputMultiOutputRegressor meta estimator

This example illustrates the use of the multioutputMultiOutputRegressor metaestimator to perform multioutput regression A random forest regressor is used which supports multioutput regression natively so the results can be compared

The random forest regressor will only ever predict values within the range of observations or closer to zero for each of the targets As a result the predictions are biased towards the centre of the circle

Using a single underlying feature the model learns both the x and y coordinate as output

1028 Chapter 5 Examples

scikitlearn user guide Release 0213
printdoc
Author Tim Head betatimgmailcom

License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearnensemble import RandomForestRegressor
from sklearnmodelselection import traintestsplit
from sklearnmultioutput import MultiOutputRegressor
Create a random dataset
rng = np.random.RandomState(1)
X = np.sort(200 * rng.rand(600, 1) - 100, axis=0)
y = np.array(np.pi * np.sin(X.ravel()) + np.cos(X.ravel()) * 5, dtype=np.float64)
X_train, X_test, y_train, y_test = traintestsplit(X, y, trainsize=400, testsize=200, randomstate=4)
maxdepth = 30
reg = MultiOutputRegressor(RandomForestRegressor(n_estimators=100, maxdepth=maxdepth, randomstate=0))
512 Ensemble methods 1029

```
scikitlearn user guide Release 0213
regmultirffitXtrain ytrain
regrrf RandomForestRegressornestimators100 maxdepthmaxdepth
randomstate2
regrrffitXtrain ytrain
Predict on new data
ymultirf regmultirfpredictXtest
yrf regrrfpredictXtest
Plot the results
pltfigure
s 50
a 04
pltscatterytest 0 ytest 1 edgecolork
cnavy ss markers alphaa labelData
pltscatterymultirf 0 ymultirf 1 edgecolork
ccornflowerblue ss alphaa
labelMulti RF score 2f regmultirfscoreXtest ytest
pltscatteryrf 0 yrf 1 edgecolork
cc ss marker alphaa
labelRF score 2f regrrfscoreXtest ytest
pltxlim6 6
pltylim6 6
pltxlabeltarget 1
pltylabeltarget 2
plttitleComparing random forests and the multioutput meta estimator
pltlegend
pltshow
Total running time of the script 0 minutes 0299 seconds
Note Click here to download the full example code
5128 Prediction Intervals for Gradient Boosting Regression
This example shows how quantile regression can be used to create prediction intervals
1030 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
import numpy as np
import matplotlib.pyplot as plt
from sklearn.ensemble import GradientBoostingRegressor
np.random.seed(1)
def fx
    The function to predict
    return x*np.sin(x)

# First the noiseless case
X = np.atleast2d(np.random.uniform(0, 100, size=100)).T
X = X.astype(np.float32)
# Observations
y = fx.ravel()
dy = 15 * 10 * np.random.randn(y.shape)
noise = np.random.randn(1, dy)
y = y.astype(np.float32)
# Mesh the input space for evaluations of the real function, the prediction and
# 512 Ensemble methods 1031
```

```
scikitlearn user guide Release 0213
its MSE
xx npatleast2dnplinspace0 10 1000T
xx xxastypenpfloat32
alpha 095
clf GradientBoostingRegressorlossquantile alphaalpha
nestimators250 maxdepth3
learningrate1 minsamplesleaf9
minsamplessplit9
clffitX y
Make the prediction on the meshed xaxis
yupper clfpredictxx
clfsetparamsalpha10 alpha
clffitX y
Make the prediction on the meshed xaxis
ylower clfpredictxx
clfsetparamslossls
clffitX y
Make the prediction on the meshed xaxis
ypred clfpredictxx
Plot the function the prediction and the 90 confidence interval based on
the MSE
fig pltfigure
pltplotxx fxx g labelrfx xsinx
pltplotX y b markersize10 labeluObservations
pltplotxx ypred r labeluPrediction
pltplotxx yupper k
pltplotxx ylower k
pltfillnpconcatenatexx xx1
npconcatenateyupper ylower1
alpha5 fcb ecNone label90 prediction interval
pltxlabelx
pltylabelfx
pltylim10 20
pltlegendlocupper left
pltshow
Total running time of the script 0 minutes 0275 seconds
Note Click here to download the full example code
5129 Gradient Boosting regularization
Illustration of the effect of different regularization strategies for Gradient Boosting The example is taken from Hastie
et al 20091
1T Hastie R Tibshirani and J Friedman “Elements of Statistical Learning Ed 2” Springer 2009
1032 Chapter 5 Examples
```

scikitlearn user guide Release 0213

The loss function used is binomial deviance Regularization via shrinkage learningrate 10 improves performance considerably In combination with shrinkage stochastic gradient boosting subsample 10 can produce more accurate models by reducing the variance via bagging Subsampling without shrinkage usually does poorly Another strategy to reduce the variance is by subsampling the features analogous to the random splits in Random Forests via the maxfeatures parameter

printdoc
Author Peter Prettenhofer peterprettenhofergmailcom

License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import ensemble
from sklearn import datasets
X y = datasets.make_hastie102(n_samples=12000, random_state=1)
X = X.astype(np.float32)
map_labels = lambda y: np.where(y == 1, 1, 0)
labels = y
np.unique(labels, return_inverse=True)
X_train, X_test, y_train, y_test = X[:2000], X[2000:], y[:2000], y[2000:]
512 Ensemble methods 1033

scikitlearn user guide Release 0213

```
ytrain ytest y2000 y2000
originalparams nestimators 1000 maxleafnodes 4 maxdepth None
randomstate 2
minsamplesplit 5
pltfigure
forlabel color setting inNo shrinkage orange
learningrate 10 subsample 10
learningrate01 turquoise
learningrate 01 subsample 10
subsample05 blue
learningrate 10 subsample 05
learningrate01 subsample05 gray
learningrate 01 subsample 05
learningrate01 maxfeatures2 magenta
learningrate 01 maxfeatures 2
params dictoriginalparams
paramsupdatesetting
clf ensembleGradientBoostingClassifier params
clffitXtrain ytrain
compute test set deviance
testdeviance npzerosparamsnestimators dtype=np.float64
fori ypred in enumerate(clf.staged_decision_function(Xtest))
    clfloss assumes that ytest[i] in {0, 1}
testdeviance[i] = clfloss(ytest[i], ypred[i])
plt.plot(npar, testdeviances, shape=(0, 15), testdeviance=5)
plt.legend(loc='upper left')
plt.xlabel('Boosting Iterations')
plt.ylabel('Test Set Deviance')
plt.show
```

Total running time of the script: 0 minutes 10180 seconds

Note: Click [here](#) to download the full example code

51210 Plot class probabilities calculated by the VotingClassifier

Plot the class probabilities of the first sample in a toy dataset predicted by three different classifiers and averaged by the VotingClassifier

First three exemplary classifiers are initialized: LogisticRegression, GaussianNB, and RandomForestClassifier, and used to initialize a softvoting VotingClassifier with weights [1, 1, 5], which means that the predicted probabilities of the RandomForestClassifier count 5 times as much as the weights of the other classifiers when the averaged probability is calculated

To visualize the probability weighting we fit each classifier on the training set and plot the predicted class probabilities

1034 Chapter 5 Examples


```
scikitlearn user guide Release 0213
for the first sample in this example dataset
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import VotingClassifier
clf1 = LogisticRegression(solver='lbfgs', max_iter=1000, random_state=123)
clf2 = RandomForestClassifier(n_estimators=100, random_state=123)
clf3 = GaussianNB()
X = np.array([10, 10, 12, 14, 34, 22, 11, 12])
y = np.array([1, 1, 2, 2])
eclf = VotingClassifier(estimators=[('clf1', clf1), ('clf2', clf2), ('clf3', clf3)],
                        voting='soft',
                        weights=[1, 1, 5])
# predict class probabilities for all classifiers
probas = clf.predict_proba(X)
# get class probabilities for the first sample in the dataset
probas[0]
```

scikitlearn user guide Release 0213

class11 pr0 0 forprinprobas

class21 pr0 1 forprinprobas

plotting

N 4 number of groups

ind nparangeN group positions

width 035 bar width

fig ax pltsubplots

bars for classifier 13

p1 axbarind nphstackclass111 0 width

colorgreen edgecolork

p2 axbarind width nphstackclass211 0 width

colorlightgreen edgecolork

bars for VotingClassifier

p3 axbarind 0 0 0 class111 width

colorblue edgecolork

p4 axbarind width 0 0 0 class211 width

colorsteelblue edgecolork

plot annotations

pltaxvline28 colork linestyledashed

axsetxticksind width

axsetxticklabelsLogisticRegression nweight 1

GaussianNB nweight 1

RandomForestClassifier nweight 5

VotingClassifier naverage probabilities

rotation40

haright

pltylim0 1

plttitleClass probabilities for sample 1 by different classifiers

pltlegendp10 p20 class 1 class 2 locupper left

plttightlayout

pltshow

Total running time of the script 0 minutes 0245 seconds

Note Click here to download the full example code

51211 Gradient Boosting regression

Demonstrate Gradient Boosting on the Boston housing dataset

This example fits a Gradient Boosting model with least squares loss and 500 regression trees of depth 4

1036 Chapter 5 Examples

scikitlearn user guide Release 0213
Out
MSE 65493
printdoc
Author Peter Prettenhofer peterprettenhofergmailcom

License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import ensemble
from sklearn import datasets
from sklearnutils import shuffle
from sklearnmetrics import meansquarederror

Load data
boston = datasets.load_boston
X, y = shuffle(boston.data, boston.target, random_state=13)
X = X.astype(np.float32)
offset = int(X.shape[0] * 0.9)
X_train, y_train, X_offset, y_offset =
X_test, y_test, X_offset, y_offset

Fit regression model
params = {'n_estimators': 500, 'max_depth': 4, 'min_samples_split': 2,
 'learning_rate': 0.01, 'loss': 'ls'}
512 Ensemble methods 1037

```
scikitlearn user guide Release 0213
clf ensembleGradientBoostingRegressor params
clffitXtrain ytrain
mse meansquarederrorytest clfpredictXtest
printMSE4f mse
```

```
Plot training deviance
compute test set deviance
testscore npzerosparamsnestimators dtype=npfloat64
for i, ypred in enumerate(clf.staged_predict(Xtest)):
    test_score[i] = clf.loss(ytest, ypred)

plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.title('Deviance')
plt.plot(nparange(params.n_estimators), 1 - clf.train_score_, 'b')
plt.xlabel('Training Set Deviance')
plt.plot(nparange(params.n_estimators), 1 - test_score, 'r')
plt.xlabel('Test Set Deviance')
plt.legend(loc='upper right')
plt.ylabel('Boosting Iterations')
plt.ylabel('Deviance')
```

```
Plot feature importance
feature_importance = clf.feature_importances_
# make importances relative to max importance
feature_importance /= feature_importance.max()
sorted_idx = np.argsort(feature_importance)
pos = np.arange(sorted_idx.shape[0], 5)
plt.subplot(1, 2, 2)
plt.barh(pos, feature_importance[sorted_idx], align='center')
plt.yticks(pos, boston.feature_names[sorted_idx])
plt.ylabel('Relative Importance')
plt.title('Variable Importance')
plt.show()
```

Total running time of the script: 0 minutes 04.16 seconds
Note: Click [here](#) to download the full example code

51212 Two-class AdaBoost

This example fits an AdaBoosted decision stump on a nonlinearly separable classification dataset composed of two “Gaussian quantiles” clusters (see `sklearn.datasets.make_gaussian_quantiles`) and plots the decision boundary and decision scores. The distributions of decision scores are shown separately for samples of class A and B. The predicted class label for each sample is determined by the sign of the decision score. Samples with decision scores greater than zero are classified as B and are otherwise classified as A. The magnitude of a decision score determines the degree of likeness with the predicted class label. Additionally, a new dataset could be constructed containing a desired purity of class B, for example, by only selecting samples with a decision score above some value.

scikitlearn user guide Release 0213
printdoc
Author Noel Dawe noeldawegmailcom

License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearnensemble import AdaBoostClassifier
from sklearn import DecisionTreeClassifier
from sklearn.datasets import makegaussianquantiles
Construct dataset
X1 y1 makegaussianquantilescov2
nsamples200 nfeatures2
nclasses2 randomstate1
X2 y2 makegaussianquantilesmean3 3 cov15
nsamples300 nfeatures2
nclasses2 randomstate1
X npconcatenateX1 X2
y npconcatenatey1 y2 1
Create and fit an AdaBoosted decision tree
bdt AdaBoostClassifierDecisionTreeClassifiermaxdepth1
algorithmSAMME
nestimators200
bdtfitX y
plotcolors br
plotstep 002
classnames AB
pltfigurefigsize10 5
512 Ensemble methods 1039

```
scikitlearn user guide Release 0213
Plot the decision boundaries
pltsubplot121
xmin xmax X 0min 1 X 0max 1
ymin ymax X 1min 1 X 1max 1
xx yy npmeshgridnparangexmin xmax plotstep
nparangeymax ymin plotstep
Z bdtpredictnpcxxravel yyravel
Z Zreshapexxshape
cs pltcontourfxx yy Z cmappltcmPaired
pltaxistight
Plot the training points
fori n c inziprange2 classnames plotcolors
idx npwherey i
pltscatterXidx 0 Xidx 1
cc cmappltcmPaired
s20 edgecolork
labelClass s n
pltxlimxmin xmax
pltylimymin ymax
pltlegendlocupper right
pltlabelx
pltlabely
plttitleDecision Boundary
Plot the twoclass decision scores
twoclassoutput bdtdecisionfunctionX
plotrange twoclassoutputmin twoclassoutputmax
pltsubplot122
fori n c inziprange2 classnames plotcolors
pltthisttwoclassoutputy i
bins10
rangeplotrange
facecolorc
labelClass s n
alpha5
edgecolork
x1 x2 y1 y2 pltaxis
pltaxisx1 x2 y1 y2 12
pltlegendlocupper right
pltlabelSamples
pltlabelScore
plttitleDecision Scores
plttightlayout
pltsubplotsadjustwspace035
pltshow
Total running time of the script 0 minutes 2255 seconds
Note Click here to download the full example code
1040 Chapter 5 Examples
```

scikitlearn user guide Release 0213
51213 OOB Errors for Random Forests

TheRandomForestClassifier is trained using bootstrap aggregation where each new tree is fit from a bootstrap sample of the training observations. The outofbag OOB error is the average error for each calculated using predictions from the trees that do not contain in their respective bootstrap sample. This allows the RandomForestClassifier to be fit and validated whilst being trained.

The example below demonstrates how the OOB error can be measured at the addition of each new tree during training. The resulting plot allows a practitioner to approximate a suitable value of nestimators at which the error stabilizes.

```
import matplotlib.pyplot as plt
from collections import OrderedDict
from sklearn.datasets import make_classification
from sklearn.ensemble import RandomForestClassifier
Author: Kian Ho huikianhogmailcom
Gilles Louppe glouppegmailcom
Andreas Mueller amuelleraisunibonnde
```

License: BSD (Binary Search Decision) 3 Clause
printdoc
1T Hastie, R Tibshirani and J Friedman "Elements of Statistical Learning Ed 2" p592-593 Springer 2009
512 Ensemble methods 1041

```
scikitlearn user guide Release 0213
RANDOMSTATE 123
Generate a binary classification dataset
X y makeclassificationnsamples500 nfeatures25
nclustersperclass1 ninformative15
randomstateRANDOMSTATE
NOTE Setting the warmstart construction parameter to True disables
support for parallelized ensembles but is necessary for tracking the OOB
error trajectory during training
ensembleclfs
RandomForestClassifier maxfeaturessqrt
RandomForestClassifiernestimators100
warmstartTrue oobscoreTrue
maxfeaturessqrt
randomstateRANDOMSTATE
RandomForestClassifier maxfeatureslog2
RandomForestClassifiernestimators100
warmstartTrue maxfeatureslog2
oobscoreTrue
randomstateRANDOMSTATE
RandomForestClassifier maxfeaturesNone
RandomForestClassifiernestimators100
warmstartTrue maxfeaturesNone
oobscoreTrue
randomstateRANDOMSTATE

Map a classifier name to a list of nestimators error rate pairs
errorrate OrderedDictlabel forlabel inensembleclfs
Range of nestimators values to explore
minestimators 15
maxestimators 175
forlabel clf inensembleclfs
foriinrangeminestimators maxestimators 1
clfsetparamsnestimatorsi
clffitX y
Record the OOB error for each nestimators setting
ooberror 1 clfoobscore
errorratelabelappendi ooberror
Generate the OOB error rate vs nestimators plot
forlabel clferr inerrorrateitems
xs ys zip clferr
pltplotxs ys labellabel
pltxlimminestimators maxestimators
pltxlabelnestimators
pltylabelOOB error rate
pltlegendlocupper right
pltshow
Total running time of the script 0 minutes 5517 seconds
1042 Chapter 5 Examples
```


scikitlearn user guide Release 0213

Note [Click here to download the full example code](#)

51214 Hashing feature transformation using Totally Random Trees

RandomTreesEmbedding provides a way to map data to a very highdimensional sparse representation which might be beneficial for classification The mapping is completely unsupervised and very efficient

This example visualizes the partitions given by several trees and shows how the transformation can also be used for nonlinear dimensionality reduction or nonlinear classification

Points that are neighboring often share the same leaf of a tree and therefore share large parts of their hashed representation This allows to separate two concentric circles simply based on the principal components of the transformed data with truncated SVD

In highdimensional spaces linear classifiers often achieve excellent accuracy For sparse binary data BernoulliNB is particularly wellsuited The bottom row compares the decision boundary obtained by BernoulliNB in the transformed space with an ExtraTreesClassifier forests learned on the original data

512 Ensemble methods 1043

```
scikitlearn user guide Release 0213
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_circles
from sklearn.ensemble import RandomTreesEmbedding
from sklearn.decomposition import TruncatedSVD
from sklearn.naive_bayes import BernoulliNB
    make a synthetic dataset
X, y = make_circles(n_samples=1000, random_state=0, noise=0.05)
    use RandomTreesEmbedding to transform data
hasher = RandomTreesEmbedding(n_estimators=10, random_state=0, max_depth=3)
X_transformed = hasher.fit_transform(X)
    Visualize result after dimensionality reduction using truncated SVD
svd = TruncatedSVD(n_components=2)
X_reduced = svd.fit_transform(X_transformed)
    Learn a Naive Bayes classifier on the transformed data
nb = BernoulliNB
nb.fit(X_transformed, y)
    Learn an ExtraTreesClassifier for comparison
trees = ExtraTreesClassifier(max_depth=3, n_estimators=10, random_state=0)
trees.fit(X, y)
    scatter plot of original and reduced data
fig, plt.subplots(figsize=(9, 8))
ax = plt.subplot(2, 2, 1)
ax.scatter(X[:, 0], X[:, 1], c=y, s=50, edgecolor='k')
ax.set_title('Original Data (2d)')
ax.set_xticks(0, 1)
ax.set_yticks(0, 1)
ax = plt.subplot(2, 2, 2)
ax.scatter(X_reduced[:, 0], X_reduced[:, 1], c=y, s=50, edgecolor='k')
ax.set_title('Truncated SVD reduction (2d) of transformed data')
X_transformed.shape[1]
ax.set_xticks(0, 1)
ax.set_yticks(0, 1)
    Plot the decision in original space. For that we will assign a color
    to each point in the mesh.
h = 0.1
x_min, x_max = X[:, 0].min() - 5, X[:, 0].max() + 5
y_min, y_max = X[:, 1].min() - 5, X[:, 1].max() + 5
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
    transform grid using RandomTreesEmbedding
transformed_grid = hasher.transform(np.c_[xx.ravel(), yy.ravel()])
y_grid_pred = nb.predict_proba(transformed_grid)[:, 1]
ax = plt.subplot(2, 2, 3)
ax.set_title('Naive Bayes on Transformed data')
1044 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
axpcolormeshxx yy ygridpredreshapexxshape
axscatterX 0 X 1 cy s50 edgcolork
axsetylim14 14
axsetxlim14 14
axsetxticks
axsetyticks
transform grid using ExtraTreesClassifier
ygridpred treespredictprobanpcxxravel yy.ravel 1
ax pltsubplot224
axsettitleExtraTrees predictions
axpcolormeshxx yy ygridpredreshapexxshape
axscatterX 0 X 1 cy s50 edgcolork
axsetylim14 14
axsetxlim14 14
axsetxticks
axsetyticks
plt.tightlayout
plt.show
Total running time of the script 0 minutes 0250 seconds
Note Click here to download the full example code
51215 Multiclass AdaBoosted Decision Trees
This example reproduces Figure 1 of Zhu et al1and shows how boosting can improve prediction accuracy on a multi
class problem The classification dataset is constructed by taking a tendimensional standard normal distribution and
defining three classes separated by nested concentric tendimensional spheres such that roughly equal numbers of
samples are in each class quantiles of the 2distribution
The performance of the SAMME and SAMMER1algorithms are compared SAMMER uses the probability estimates
to update the additive model while SAMME uses the classifications only As the example illustrates the SAMMER
algorithm typically converges faster than SAMME achieving a lower test error with fewer boosting iterations The
error of each algorithm on the test set after each boosting iteration is shown on the left the classification error on the
test set of each tree is shown in the middle and the boost weight of each tree is shown on the right All trees have a
weight of one in the SAMMER algorithm and therefore are not shown
1
10 Zhu H Zou S Rosset T Hastie “Multiclass AdaBoost” 2009
512 Ensemble methods 1045
```

scikitlearn user guide Release 0213
printdoc
Author Noel Dawe noeldawegmailcom

License BSD (Binary Search Decision) 3 clause
import matplotlib.pyplot as plt
from sklearn.datasets import makegaussianquantiles
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import accuracyscore
from sklearn.tree import DecisionTreeClassifier
X y makegaussianquantilesnsamples13000 nfeatures10
nclasses3 randomstate1
nsplit 3000
Xtrain Xtest Xnsplit Xnsplit
ytrain ytest ynsplit ynsplit
bdtreal AdaBoostClassifier
DecisionTreeClassifiermaxdepth2
nestimators600
learningrate1
bdt discrete AdaBoostClassifier
DecisionTreeClassifiermaxdepth2
nestimators600
learningrate15
algorithmSAMME
bdtrealfitXtrain ytrain
bdt discretefitXtrain ytrain
realtesterrors
discretetesterrors
forrealtestpredict discretetrainpredict inzip
bdtrealstagedpredictXtest bdt discretetestagedpredictXtest
realtesterrorsappend
1 accuracyscorerealtestpredict ytest
1046 Chapter 5 Examples

```
scikitlearn user guide Release 0213
discretetesterrorsappend
1 accuracyscorediscretetrainpredict ytest
ntreesdiscrete lenbdttdiscrete
ntreesreal lenbdtreal
Boosting might terminate early but the following arrays are always
nestimators long We crop them to the actual number of trees here
discreteestimatorerrors bdttdiscreteestimatorerrorsntreesdiscrete
realestimatorerrors bdtrealstimatorerrorsntreesreal
discreteestimatorweights bdttdiscreteestimatorweightsntreesdiscrete
pltfigurefigsize15 5
pltsubplot131
pltplotrange1 ntreesdiscrete 1
discretetesterrors cblack labelSAMME
pltplotrange1 ntreesreal 1
realtesterrors cblack
linestyledashed labelSAMMER
pltlegend
pltylim018 062
pltlabelTest Error
pltxlabelNumber of Trees
pltsubplot132
pltplotrange1 ntreesdiscrete 1 discreteestimatorerrors
b labelSAMME alpha5
pltplotrange1 ntreesreal 1 realestimatorerrors
r labelSAMMER alpha5
pltlegend
pltlabelError
pltxlabelNumber of Trees
pltylim2
maxrealestimatorerrorsmax
discreteestimatorerrorsmax 12
pltxlim20 lenbdttdiscrete 20
pltsubplot133
pltplotrange1 ntreesdiscrete 1 discreteestimatorweights
b labelSAMME
pltlegend
pltlabelWeight
pltxlabelNumber of Trees
pltylim0 discreteestimatorweightsmax 12
pltxlim20 ntreesdiscrete 20
prevent overlapping yaxis labels
pltsubplotsadjustwspace025
pltshow
Total running time of the script 0 minutes 11401 seconds
Note Click here to download the full example code
512 Ensemble methods 1047
```

scikitlearn user guide Release 0213

51216 Discrete versus Real AdaBoost

This example is based on Figure 102 from Hastie et al 2009¹ and illustrates the difference in performance between the discrete SAMME2 boosting algorithm and real SAMMER boosting algorithm. Both algorithms are evaluated on a binary classification task where the target Y is a nonlinear function of 10 input features.

Discrete SAMME AdaBoost adapts based on errors in predicted class labels, whereas real SAMMER uses the predicted class probabilities.

printdoc

Author Peter Prettenhofer peterprettenhofergmailcom

Noel Dawe noeldawegmailcom

License BSD (Binary Search Decision) 3 clause

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import zero_one_loss

¹T Hastie, R Tibshirani and J Friedman “Elements of Statistical Learning Ed 2” Springer 2009

2

¹⁰Zhu H, Zou S, Rosset T, Hastie “Multiclass AdaBoost” 2009

1048 Chapter 5 Examples

```
scikitlearn user guide Release 0213
from sklearnensemble import AdaBoostClassifier
nestimators 400
A learning rate of 1 may not be optimal for both SAMME and SAMMER
learningrate 1
X y datasetsmakehastie102nsamples12000 randomstate1
Xtest ytest X2000 y2000
Xtrain ytrain X2000 y2000
dtstump DecisionTreeClassifiermaxdepth1 minsamplesleaf1
dtstumpfitXtrain ytrain
dtstumperr 10 dtstumpscoreXtest ytest
dt DecisionTreeClassifiermaxdepth9 minsamplesleaf1
dtfitXtrain ytrain
dterr 10 dtscoreXtest ytest
adadiscrete AdaBoostClassifier
baseestimator dtstump
learningrate learningrate
nestimators nestimators
algorithm SAMME
adadiscretefitXtrain ytrain
adareal AdaBoostClassifier
baseestimator dtstump
learningrate learningrate
nestimators nestimators
algorithm SAMMER
adarealfitXtrain ytrain
fig pltfigure
ax figaddsubplot111
axplot1 nestimators dtstumperr 2 k
labelDecision Stump Error
axplot1 nestimators dterr 2 k
labelDecision Tree Error
adadiscreteerr npzerosnestimators
fori ypred in enumerate(adadiscretestagedpredictXtest
adadiscreteerrri zeroonelosypred ytest
adadiscreteerrtrain npzerosnestimators
fori ypred in enumerate(adadiscretestagedpredictXtrain
adadiscreteerrtraini zeroonelosypred ytrain
adarealerr npzerosnestimators
fori ypred in enumerate(adarealstagedpredictXtest
adarealerrri zeroonelosypred ytest
adarealerrtrain npzerosnestimators
fori ypred in enumerate(adarealstagedpredictXtrain
adarealerrtraini zeroonelosypred ytrain
512 Ensemble methods 1049
```

```
scikitlearn user guide Release 0213
axplotnparangenestimators 1 adadiscreteerr
labelDiscrete AdaBoost Test Error
colorred
axplotnparangenestimators 1 adadiscreteerrtrain
labelDiscrete AdaBoost Train Error
colorblue
axplotnparangenestimators 1 adarealerr
labelReal AdaBoost Test Error
colororange
axplotnparangenestimators 1 adarealerrtrain
labelReal AdaBoost Train Error
colorgreen
axsetylim00 05
axsetxlabelnestimators
axsetylabelerror rate
leg axlegendlocupper right fancyboxTrue
leggetframesetalpha07
pltshow
Total running time of the script 0 minutes 4579 seconds
Note Click here to download the full example code
51217 Early stopping of Gradient Boosting
Gradient boosting is an ensembling technique where several weak learners regression trees are combined to yield a
powerful single model in an iterative fashion
Early stopping support in Gradient Boosting enables us to find the least number of iterations which is sufficient to
build a model that generalizes well to unseen data
The concept of early stopping is simple We specify a validationfraction which denotes the fraction of the
whole dataset that will be kept aside from training to assess the validation loss of the model The gradient boosting
model is trained using the training set and evaluated using the validation set When each additional stage of regression
tree is added the validation set is used to score the model This is continued until the scores of the model in the last
niternochange stages do not improve by atleast tol After that the model is considered to have converged
and further addition of stages is “stopped early”
The number of stages of the final model is available at the attribute nestimators
This example illustrates how the early stopping can used in the sklearnensemble
GradientBoostingClassifier model to achieve almost the same accuracy as compared to a model
built without early stopping using many fewer estimators This can significantly reduce training time memory usage
and prediction latency
Authors Vighnesh Birodkar vighneshbirodkarnyuedu
Raghav RV rvraghav93gmailcom
License BSD (Binary Search Decision) 3 clause
import time
import numpy as np
1050 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
import matplotlib.pyplot as plt
from sklearn import ensemble
from sklearn import datasets
from sklearn.model_selection import train_test_split
print doc
data_list datasets load iris datasets load digits
data_list ddata dtarget ford in data list
data_list datasets make hastie 102
names Iris Data Digits Data Hastie Data
ngb
score gb
time gb
ng bes
score gb bes
time gb bes
nestimators 500
for X y in data_list
X_train X_test y_train y_test train_test_split X y test_size 0.2
random_state 0
We specify that if the scores dont improve by atleast 0.01 for the last
10 stages stop fitting additional stages
gbes ensemble Gradient Boosting Classifier nestimators nestimators
validation_fraction 0.2
n_iter_no_change 5 tol 0.01
random_state 0
gb ensemble Gradient Boosting Classifier nestimators nestimators
random_state 0
start time
time gb append time start
start time
gbes fit X_train y_train
time gb bes append time start
score gb append gb score X_test y_test
score gb bes append gb score X_test y_test
ngb append gb nestimators
ng bes append gb bes nestimators
barwidth 0.2
n len data_list
index nparange 0 n barwidth barwidth 25
index index 0 n
512 Ensemble methods 1051
```

```
scikitlearn user guide Release 0213
Compare scores with and without early stopping
pltfigurefigsize9 5
bar1 pltbarindex scoregb barwidth labelWithout early stopping
colorcrimson
bar2 pltbarindex barwidth scoregbes barwidth
labelWith early stopping colorcoral
pltxticksindex barwidth names
plttyticksnparange0 13 01
defaultlabelrects nestimators
```

Attach a text label above each bar displaying nestimators of each model

```
fori rect inenumeratorects
plttextrectgetx rectgetwidth 2
105rectgetheight nest d nestimatorsi
hacenter vabottom
autolabelbar1 ngb
autolabelbar2 ngbes
pltylim0 13
pltlegendlocbest
pltgridTrue
pltxlabelDatasets
pltylabelTest score
pltshow
1052 Chapter 5 Examples
```

scikitlearn user guide Release 0213
Compare fit times with and without early stopping
pltfigurefigsize9 5
bar1 pltbarindex timegb barwidth labelWithout early stopping
colorcrimson
bar2 pltbarindex barwidth timegbes barwidth
labelWith early stopping colorcoral
maxy npamaxnpmaximumtimegb timegbes
pltxticksindex barwidth names
pltyticksnplinspace0 13 maxy 13
autolabelbar1 ngb
autolabelbar2 ngbes
pltylim0 13 maxy
pltlegendlocbest
pltgridTrue
pltxlabelDatasets
pltylabelFit Time
pltshow
512 Ensemble methods 1053

scikitlearn user guide Release 0213

Total running time of the script 0 minutes 15622 seconds

Note [Click here to download the full example code](#)

51218 Feature transformations with ensembles of trees

Transform your features into a higher dimensional sparse space Then train a linear model on these features

First fit an ensemble of trees totally random trees a random forest or gradient boosted trees on the training set Then each leaf of each tree in the ensemble is assigned a fixed arbitrary feature index in a new feature space These leaf indices are then encoded in a onehot fashion

Each sample goes through the decisions of each tree of the ensemble and ends up in one leaf per tree The sample is encoded by setting feature values for these leaves to 1 and the other feature values to 0

The resulting transformer has then learned a supervised sparse highdimensional categorical embedding of the data

1054 Chapter 5 Examples

scikitlearn user guide Release 0213

-

512 Ensemble methods 1055

scikitlearn user guide Release 0213

- Author Tim Head betatimgmailcom

License BSD (Binary Search Decision) 3 clause
import numpy as np
nprandomseed10
import matplotlib.pyplot as plt
from sklearn.datasets import makeclassification
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
GradientBoostingClassifier
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_curve
from sklearn.pipeline import make_pipeline
nestimator 10
X y makeclassificationnnsamples80000
Xtrain Xtest ytrain ytest train_test_splitX y testsize05
It is important to train the ensemble of trees on a different subset
of the training data than the linear regression model to avoid
overfitting in particular if the total number of leaves is
similar to the number of training samples
Xtrain Xtrainlr ytrain ytrainlr train_test_split
1056 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Xtrain ytrain testsize05
  Unsupervised transformation based on totally random trees
rt RandomTreesEmbeddingmaxdepth3 nestimatorsnestimator
randomstate0
rtlm LogisticRegressionsolverlbfgs maxiter1000
pipeline makepipelinert rtlm
pipelinefitXtrain ytrain
ypredrt pipelinepredictprobaXtest 1
fprrtlm tprrtlm roccurveystest ypredrt
  Supervised transformation based on random forests
rf RandomForestClassifiermaxdepth3 nestimatorsnestimator
rfenc OneHotEncodercategoriesauto
rflm LogisticRegressionsolverlbfgs maxiter1000
rffitXtrain ytrain
rfencfitrfapplyXtrain
rflmfitrfenctransformrfapplyXtrainlr ytrainlr
ypredrflm rflmpredictprobarfencctransformrfapplyXtest 1
fprrrflm tprrrflm roccurveystest ypredrflm
  Supervised transformation based on gradient boosted trees
grd GradientBoostingClassifiernestimatorsnestimator
grdenc OneHotEncodercategoriesauto
grdlm LogisticRegressionsolverlbfgs maxiter1000
grdfitXtrain ytrain
grdencfitgrdapplyXtrain 0
grdlmfitgrdenctransformgrdapplyXtrainlr 0 ytrainlr
ypredgrdlm grdlmpredictproba
grdenctransformgrdapplyXtest 0 1
fprgrdlm tprgrdlm roccurveystest ypredgrdlm
  The gradient boosted model by itself
ypredgrd grdpredictprobaXtest 1
fprgrd tprgrd roccurveystest ypredgrd
  The random forest model by itself
ypredrf rfpredictprobaXtest 1
fprrf tprrf roccurveystest ypredrf
pltfigure1
pltplot0 1 0 1 k
pltplotfprrtlm tprrtlm labelIRT LR
pltplotfprrf tprrf labelRF
pltplotfprrrflm tprrrflm labelRF LR
pltplotfprgrd tprgrd labelGBT
pltplotfprgrdlm tprgrdlm labelGBT LR
pltxlabelFalse positive rate
pltylabelTrue positive rate
plttitleROC curve
pltlegendlocbest
pltshow
pltfigure2
pltxlim0 02
512 Ensemble methods 1057
```

scikitlearn user guide Release 0213

```
pltylim(0, 1)
pltplot(0, 1, 0, 1, k)
pltplotfprtlm tprtlm label=RT_LR
pltplotfprf tprf label=RF
pltplotfprflm tprflm label=RF_LR
pltplotfprgrd tprgrd label=GBT
pltplotfprgrdlm tprgrdlm label=GBT_LR
pltxlabel False positive rate
pltylabel True positive rate
plttitle ROC curve zoomed in at top left
pltlegend loc=best
pltshow
```

Total running time of the script: 0 minutes 2261 seconds

Note: Click [here](#) to download the full example code

5.12.19 Gradient Boosting Out-of-Bag estimates

Out-of-bag (OOB) estimates can be a useful heuristic to estimate the “optimal” number of boosting iterations. OOB estimates are almost identical to cross-validation estimates but they can be computed on-the-fly without the need for repeated model fitting. OOB estimates are only available for Stochastic Gradient Boosting (ie. subsample = 1). The estimates are derived from the improvement in loss based on the examples not included in the bootstrap sample (the so-called out-of-bag examples). The OOB estimator is a pessimistic estimator of the true test loss but remains a fairly good approximation for a small number of trees.

The figure shows the cumulative sum of the negative OOB improvements as a function of the boosting iteration. As you can see, it tracks the test loss for the first hundred iterations but then diverges in a pessimistic way. The figure also shows the performance of 3-fold cross-validation, which usually gives a better estimate of the test loss but is computationally more demanding.


```
scikitlearn user guide Release 0213
Out
Accuracy 06840
printdoc
Author Peter Prettenhofer peterprettenhofergmailcom

License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import ensemble
from sklearn.model_selection import KFold
from sklearn.model_selection import train_test_split
from scipy.special import expit
Generate data adapted from G Ridgeway's gbm example
nsamples 1000
512 Ensemble methods 1059
```

```
scikitlearn user guide Release 0213
randomstate np.random.RandomState(13)
x1 randomstate.uniform(size=samples)
x2 randomstate.uniform(size=samples)
x3 randomstate.randint(0, 4, size=samples)
p expit(np.sin(3 * x1 - 4 * x2 - x3))
y randomstate.binomial(1, p, size=samples)
X np.c_[x1, x2, x3]
X X.astype(np.float32)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5,
random_state=9)
# Fit classifier with out-of-bag estimates
params = {'n_estimators': 1200, 'max_depth': 3, 'subsample': 0.5,
learning_rate': 0.01, 'min_samples_leaf': 1, 'random_state': 3}
clf = ensemble.GradientBoostingClassifier(params)
clf.fit(X_train, y_train)
acc = clf.score(X_test, y_test)
print('Accuracy: %f' % acc)
n_estimators = params['n_estimators']
x = np.arange(n_estimators)
def heldout_score(clf, X_test, y_test):
    # Compute deviance scores on X_test and y_test
    score = np.zeros(n_estimators, dtype=np.float64)
    for i, y_pred in enumerate(clf.staged_decision_function(X_test)):
        score[i] = clf.loss(y_test, y_pred)
    return score
def cv_estimate(n_splits=None):
    cv = KFold(n_splits=n_splits)
    cv_clf = ensemble.GradientBoostingClassifier(params)
    val_scores = np.zeros(n_estimators, dtype=np.float64)
    for train, test in cv.split(X_train, y_train):
        cv_clf.fit(X_train[train], y_train[train])
        val_scores[test] = heldout_score(cv_clf, X_train[test], y_train[test])
    return val_scores
# Estimate best estimator using cross-validation
cv_score = cv_estimate(3)
# Compute best estimator for test data
test_score = heldout_score(clf, X_test, y_test)
# Negative cumulative sum of OOB improvements
cumsum = np.cumsum(cv_loss_improvement)
# Min loss according to OOB
oob_best_iter = np.argmin(cumsum)
1060 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
min loss according to test normalize such that first loss is 0
testscore testscore0
testbestiter xnpargmintestscore
min loss according to cv normalize such that first loss is 0
cvscore cvscore0
cvbestiter xnpargmincvscore
color brew for the three curves
oobcolor listmap lambdax x 2560 190 174 212
testcolor listmap lambdax x 2560 127 201 127
cvcolor listmap lambdax x 2560 253 192 134
plot curves and vertical lines for best iterations
pltplotx cumsum labelOOB loss coloroobcolor
pltplotx testscore labelTest loss colortestcolor
pltplotx cvscore labelCV loss colorcvcvcolor
pltaxvlinexoobbestiter coloroobcolor
pltaxvlinextestbestiter colortestcolor
pltaxvlinexcvbestiter colorcvcvcolor
add three vertical lines to xticks
xticks pltxticks
xtickspos nparrayxticks0tolist
oobbestiter cvbestiter testbestiter
xtickslabel nparraylistmap lambdat intt xticks0
OOB CV (Cross-Validation) Test
ind npargsortxtickspos
xtickspos xticksposind
xtickslabel xtickslabelind
pltxticksxtickspos xtickslabel
pltlegendlocupper right
pltlabelnormalized loss
pltlabelnumber of iterations
pltshow
Total running time of the script 0 minutes 2759 seconds
Note Click here to download the full example code
51220 Single estimator versus bagging biasvariance decomposition
This example illustrates and compares the biasvariance decomposition of the expected mean squared error of a single
estimator against a bagging ensemble
In regression the expected mean squared error of an estimator can be decomposed in terms of bias variance and
noise On average over datasets of the regression problem the bias term measures the average amount by which the
predictions of the estimator differ from the predictions of the best possible estimator for the problem ie the Bayes
model The variance term measures the variability of the predictions of the estimator when fit over different instances
LS of the problem Finally the noise measures the irreducible part of the error which is due the variability in the data
The upper left figure illustrates the predictions in dark red of a single decision tree trained over a random dataset LS
the blue dots of a toy 1d regression problem It also illustrates the predictions in light red of other single decision
512 Ensemble methods 1061
```

scikitlearn user guide Release 0213

trees trained over other and different randomly drawn instances LS of the problem Intuitively the variance term here corresponds to the width of the beam of predictions in light red of the individual estimators The larger the variance the more sensitive are the predictions for xto small changes in the training set The bias term corresponds to the difference between the average prediction of the estimator in cyan and the best possible model in dark blue On this problem we can thus observe that the bias is quite low both the cyan and the blue curves are close to each other while the variance is large the red beam is rather wide

The lower left figure plots the pointwise decomposition of the expected mean squared error of a single decision tree It confirms that the bias term in blue is low while the variance is large in green It also illustrates the noise part of the error which as expected appears to be constant and around 001

The right figures correspond to the same plots but using instead a bagging ensemble of decision trees In both figures we can observe that the bias term is larger than in the previous case In the upper right figure the difference between the average prediction in cyan and the best possible model is larger eg notice the offset around x2 In the lower right figure the bias curve is also slightly higher than in the lower left figure In terms of variance however the beam of predictions is narrower which suggests that the variance is lower Indeed as the lower right figure confirms the variance term in green is lower than for single decision trees Overall the bias variance decomposition is therefore no longer the same The tradeoff is better for bagging averaging several decision trees fit on bootstrap copies of the dataset slightly increases the bias term but allows for a larger reduction of the variance which results in a lower overall mean squared error compare the red curves int the lower figures The script output also confirms this intuition The total error of the bagging ensemble is lower than the total error of a single decision tree and this difference indeed mainly stems from a reduced variance

For further details on biasvariance decomposition see section 73 of1

1T Hastie R Tibshirani and J Friedman “Elements of Statistical Learning” Springer 2009

1062 Chapter 5 Examples

scikitlearn user guide Release 0213

References

Out

Tree 00255 error 00003 bias2 00152 var 00098 noise

BaggingTree 00196 error 00004 bias2 00092 var 00098 noise

printdoc

Author Gilles Louppe glouppegmailcom

License BSD (Binary Search Decision) 3 clause

import numpy as np

import matplotlib.pyplot as plt

from sklearnensemble import BaggingRegressor

from sklearnntree import DecisionTreeRegressor

512 Ensemble methods 1063

scikitlearn user guide Release 0213

Settings
nrepeat 50 Number of iterations for computing expectations
ntrain 50 Size of the training set
ntest 1000 Size of the test set
noise 01 Standard deviation of the noise
nprandomseed0

Change this for exploring the biasvariance decomposition of other
estimators This should work well for estimators with high variance eg
decision trees or KNN but poorly for estimators with low variance eg
linear models

estimators Tree DecisionTreeRegressor
BaggingTree BaggingRegressorDecisionTreeRegressor
nestimators lenestimators

Generate data

```
def fx
x x.ravel
return npexp(2 * 15 * npexp(2 * 2))
def generate_samples(noise, nrepeat):
X = np.random.randn(samples, 10) * 5
X = np.sort(X)
if nrepeat == 1:
y = fx(X)
else:
y = np.zeros((samples, nrepeat))
for i in range(nrepeat):
y[i] = fx(np.random.randn(100, noise, samples))
X = X.reshape(samples, 1)
return X, y
X_train, y_train = generate_samples(ntrain, noise, noise)
X_train.append(X)
y_train.append(y)
X_test, y_test = generate_samples(ntest, noise, noise, nrepeat, nrepeat)
plt.figure(figsize=(10, 8))
# Loop over estimators to compare
for name, estimator in enumerate(estimators):
# Compute predictions
y_predict = np.zeros(ntest)
nrepeat
```

1064 Chapter 5 Examples

```
scikitlearn user guide Release 0213
foriinrangenrepeat
estimatorfitXtraini ytraini
ypredict i estimatorpredictXtest
  Bias2 Variance Noise decomposition of the mean squared error
yerror npzerosntest
foriinrangenrepeat
forjinrangenrepeat
yerror ytest j ypredict i 2
yerror nrepeat nrepeat
ynoise npvarytest axis1
ybias fXtest npmeanypredict axis1 2
yvar npvarypredict axis1
print0 14f error 24f bias2
  34f var 44f noiseformatname
npmeanerror
npmeanbias
npmeanvar
npmeannoise
  Plot figures
pltsubplot2 nestimators n 1
pltplotXtest fXtest b labelfx
pltplotXtrain0 ytrain0 b labelLS y fxnoise
foriinrangenrepeat
ifi 0
pltplotXtest ypredict i r labelryx
else
pltplotXtest ypredict i r alpha005
pltplotXtest npmeanypredict axis1 c
labelrmathbbELS yx
pltxlim5 5
plttitlename
ifn nestimators 1
pltlegendloc11 5
pltsubplot2 nestimators nestimators n 1
pltplotXtest yerror r labelerrorx
pltplotXtest ybias b labelbias2x
pltplotXtest yvar g labelvariancex
pltplotXtest ynoise c labelnoisex
pltxlim5 5
pltylim0 01
ifn nestimators 1
pltlegendloc11 5
pltsubplotsadjustright75
512 Ensemble methods 1065
```

scikitlearn user guide Release 0213

pltshow

Total running time of the script 0 minutes 0515 seconds

Note Click here to download the full example code

51221 Plot the decision surfaces of ensembles of trees on the iris dataset

Plot the decision surfaces of forests of randomized trees trained on pairs of features of the iris dataset

This plot compares the decision surfaces learned by a decision tree classifier first column by a random forest classifier second column by an extra trees classifier third column and by an AdaBoost classifier fourth column

In the first row the classifiers are built using the sepal width and the sepal length features only on the second row using the petal length and sepal length only and on the third row using the petal width and the petal length only

In descending order of quality when trained outside of this example on all 4 features using 30 estimators and scored using 10 fold cross validation we see

ExtraTreesClassifier 095 score

RandomForestClassifier 094 score

AdaBoostDecisionTreemaxdepth3 094 score

DecisionTreemaxdepth None 094 score

Increasing maxdepth for AdaBoost lowers the standard deviation of the scores but the average score does not improve

See the console's output for further details about each model

In this example you might try to

1 vary the maxdepth for the DecisionTreeClassifier andAdaBoostClassifier perhaps try maxdepth3 for theDecisionTreeClassifier ormaxdepthNone for

AdaBoostClassifier

2 varynestimators

It is worth noting that RandomForests and ExtraTrees can be fitted in parallel on many cores as each tree is built independently of the others AdaBoost's samples are built sequentially and so do not use multiple cores

1066 Chapter 5 Examples


```
scikitlearn user guide Release 0213
Out
DecisionTree with features 0 1 has a score of 092666666666666666
RandomForest with 30 estimators with features 0 1 has a score of 092666666666666666
ExtraTrees with 30 estimators with features 0 1 has a score of 092666666666666666
AdaBoost with 30 estimators with features 0 1 has a score of 084
DecisionTree with features 0 2 has a score of 099333333333333333
RandomForest with 30 estimators with features 0 2 has a score of 099333333333333333
ExtraTrees with 30 estimators with features 0 2 has a score of 099333333333333333
AdaBoost with 30 estimators with features 0 2 has a score of 099333333333333333
DecisionTree with features 2 3 has a score of 099333333333333333
RandomForest with 30 estimators with features 2 3 has a score of 099333333333333333
ExtraTrees with 30 estimators with features 2 3 has a score of 099333333333333333
AdaBoost with 30 estimators with features 2 3 has a score of 099333333333333333
printdoc
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
512 Ensemble methods 1067
```

```
scikitlearn user guide Release 0213
from sklearn.datasets import loadiris
from sklearn.ensemble import RandomForestClassifier ExtraTreesClassifier
AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
Parameters
nclasses 3
nestimators 30
cmap plt.cm.RdYlBu
plotstep 0.02 fine step width for decision surface contours
plotstepcoarser 0.05 step widths for coarse classifier guesses
RANDOMSEED 13 fix the seed on each iteration
Load data
iris loadiris
plotidx 1
models DecisionTreeClassifiermaxdepth=None
RandomForestClassifiernestimators=nestimators
ExtraTreesClassifiernestimators=nestimators
AdaBoostClassifierDecisionTreeClassifiermaxdepth=3
nestimators=nestimators
for pair in 0 1 0 2 2 3
formodel in models
    We only take the two corresponding features
X iris.data pair
y iris.target
Shuffle
idx np.arange(X.shape[0])
np.random.seed(RANDOMSEED)
np.random.shuffle(idx)
X X[idx]
y y[idx]
Standardize
mean X.mean(axis=0)
std X.std(axis=0)
X X - mean std
Train
model.fit(X, y)
scores = model.score(X, y)
Create a title for each column and the console by using str and
slicing away useless parts of the string
model_title = str(type(model)) + split
12 len(Classifier)
model_details = model_title
if hasattr(model, 'estimators'):
    model_details += with_estimators.format
len(model.estimators)
print(model_details with features pair)
has a score of scores
1068 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
plt.subplot(3, 4, plotidx)
if plotidx < len(models):
    Add a title at the top of each column
    plt.title(model.title, fontsize=9)
    Now plot the decision boundary using a fine mesh as input to a
    filled contour plot
    xmin, xmax, X0min, 1, X0max, 1
    ymin, ymax, X1min, 1, X1max, 1
    xx, yy = np.meshgrid(np.arange(xmin, xmax, plotstep),
    np.arange(ymin, ymax, plotstep))
    Plot either a single DecisionTreeClassifier or alpha blend the
    decision surfaces of the ensemble of classifiers
    if isinstance(model, DecisionTreeClassifier):
        Z = model.predict(np.c_[x.ravel(), y.ravel()])
        Z = Z.reshape(xx.shape)
        cs = plt.contourf(xx, yy, Z, cmap=cmap)
    else:
        Choose alpha blend level with respect to the number
        of estimators
        that are in use, noting that AdaBoost can use fewer estimators
        than its maximum if it achieves a good enough fit early on
        estimator_alpha = 10 / len(model.estimators)
        for tree in model.estimators:
            Z = tree.predict(np.c_[x.ravel(), y.ravel()])
            Z = Z.reshape(xx.shape)
            cs = plt.contourf(xx, yy, Z * estimator_alpha, cmap=cmap)
        Build a coarser grid to plot a set of ensemble classifications
        to show how these are different to what we see in the decision
        surfaces. These points are regularly spaced and do not have a
        black outline
        xxcoarser, yycoarser = np.meshgrid(
            np.arange(xmin, xmax, plotstep * coarser),
            np.arange(ymin, ymax, plotstep * coarser))
        Zpointscoarser = model.predict(np.c_[xxcoarser.ravel(),
        yycoarser.ravel()])
        reshape_xxcoarser_shape =
        cspoints = plt.scatter(xxcoarser, yycoarser, s=15,
        c=Zpointscoarser, cmap=cmap,
        edgecolor=None)
        Plot the training points; these are clustered together and have a
        black outline
        plt.scatter(X0, X1, y,
            cmap=ListedColorMap(r, y, b),
            edgecolor=k, s=20)
        plotidx += 1 # move on to the next plot in sequence
        plt.suptitle('Classifiers on feature subsets of the Iris dataset',
        fontsize=12)
        plt.tight_layout()
        plt.tight_layout(pad=0.2, w_pad=0.2, pad=2.5)
        plt.show()
Total running time of the script: 0 minutes 61.77 seconds
512 Ensemble methods 1069
```

scikitlearn user guide Release 0213

513 Tutorial exercises

Exercises for the tutorials

Note Click here to download the full example code

5131 Digits Classification Exercise

A tutorial exercise regarding the use of classification techniques on the Digits dataset

This exercise is used in the Classification part of the Supervised learning predicting an output variable from high dimensional observations section of the A tutorial on statistical learning for scientific data processing

Out

KNN score 0961111

LogisticRegression score 0933333

printdoc

from sklearn import datasets neighbors linearmodel

digits datasetsloadaddigits

Xdigits digitsdata digitsdatamax

ydigits digitstarget

nsamples lenXdigits

Xtrain Xdigitsint9 nsamples

ytrain ydigitsint9 nsamples

Xtest Xdigitsint9 nsamples

ytest ydigitsint9 nsamples

knn neighborsKNeighborsClassifier

logistic linearmodelLogisticRegressionssolverlbfgs maxiter1000

multiclassmultinomial

printKNN score f knnfitXtrain ytrainscoreXtest ytest

printLogisticRegression score f

logisticfitXtrain ytrainscoreXtest ytest

Total running time of the script 0 minutes 0432 seconds

Note Click here to download the full example code

1070 Chapter 5 Examples

```
scikitlearn user guide Release 0213
5132 Crossvalidation on Digits Dataset Exercise
A tutorial exercise using Crossvalidation with an SVM (Support Vector Machine) on the Digits dataset
This exercise is used in the Crossvalidation generators part of the Model selection choosing estimators and their
parameters section of the A tutorial on statisticallearning for scientific data processing
printdoc
import numpy as np
from sklearnmodelselection import crossvalscore
from sklearn import datasets svm
digits datasetsloaddigits
X digitsdata
y digitstarget
svc svmSVCKernellinear
Cs nplogspace10 0 10
scores list
scoresstd list
forCinCs
svcC C
thisscores crossvalscoresvc X y cv5 njobs1
513 Tutorial exercises 1071
```

scikitlearn user guide Release 0213

scoresappendnpmeanthisscores

scoresstdappendnpstdthisscores

Do the plotting

import matplotlib.pyplot as plt

pltfigure

pltsemilogxCs scores

pltsemilogxCs nparrayscores nparrayscoresstd b

pltsemilogxCs nparrayscores nparrayscoresstd b

locs labels pltxticks

pltxtickslocs listmap lambdax g x locs

pltlabelCV score

pltlabelParameter C

pltylim0 11

pltshow

Total running time of the script 0 minutes 8826 seconds

Note Click here to download the full example code

5133 SVM (Support Vector Machine) Exercise

A tutorial exercise for using different SVM (Support Vector Machine) kernels

This exercise is used in the Using kernels part of the Supervised learning predicting an output variable from high dimensional observations section of the A tutorial on statisticallearning for scientific data processing

1072 Chapter 5 Examples

scikitlearn user guide Release 0213

-

513 Tutorial exercises 1073

scikitlearn user guide Release 0213

```
•
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets svm
iris = datasets.load_iris
X = iris.data
y = iris.target
X = X[y == 0, :]
y = y[y == 0]
n_sample = len(X)
np.random.seed(0)
order = np.random.permutation(n_sample)
X = X[order]
y = y[order]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
# fit the model
513 Tutorial exercises 1075
```

scikitlearn user guide Release 0213
forkernelinlinear rbf poly
clf svmSVCKernelkernel gamma10
clffitXtrain ytrain
pltfigure
pltclf
pltscatterX 0 X 1 cy zorder10 cmappltcmPaired
edgecolor k s20
Circle out the test data
pltscatterXtest 0 Xtest 1 s80 facecolorsnone
zorder10 edgecolor k
pltaxis tight
xmin X 0min
xmax X 0max
ymin X 1min
ymax X 1max
XX YY npmgridxminxmax200j yminymax200j
Z clfdecisionfunctionnpcXXravel YYravel
Put the result into a color plot
Z ZreshapeXXshape
pltcolormeshXX YY Z 0 cmappltcmPaired
pltcontourXX YY Z colorsk k k
linestyles levels5 0 5
plttitlekernel
pltshow
Total running time of the script 0 minutes 5320 seconds
Note Click here to download the full example code
5134 Crossvalidation on diabetes Dataset Exercise
A tutorial exercise which uses crossvalidation with linear models
This exercise is used in the Crossvalidated estimators part of the Model selection choosing estimators and their
parameters section of the A tutorial on statistical learning for scientific data processing
1076 Chapter 5 Examples

scikitlearn user guide Release 0213

Out

Answer to the bonus question how much can you trust the selection of alpha
Alpha parameters maximising the generalization score on different
subsets of the data

fold 0 alpha 005968 score 054209

fold 1 alpha 004520 score 015523

fold 2 alpha 007880 score 045193

Answer Not very much since we obtained different alphas for different
subsets of the data and moreover the scores for these alphas differ
quite substantially

printdoc

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.linear_model import LassoCV

513 Tutorial exercises 1077

```
scikitlearn user guide Release 0213
from sklearn.linear_model import Lasso
from sklearn.model_selection import KFold
from sklearn.model_selection import GridSearchCV
diabetes = datasets.load_diabetes
X = diabetes.data[150:]
y = diabetes.target[150:]
lasso = Lasso(random_state=0, max_iter=10000)
alphas = np.logspace(-4, 0, 30)
tuned_parameters = {'alpha': alphas}
n_folds = 5
clf = GridSearchCV(lasso, tuned_parameters, cv=kf, refit=False)
clf.fit(X, y)
scores = clf.cv_results_['mean_test_score']
scores_std = clf.cv_results_['std_test_score']
plt.figure(figsize=(8, 6))
plt.semilogx(alphas, scores)
    plot error lines showing std errors of the scores
std_err = scores_std / np.sqrt(n_folds)
plt.semilogx(alphas, scores, std_err, b)
plt.semilogx(alphas, scores, std_err, b)
    alpha=0.2 controls the translucency of the fill color
plt.fill_between(alphas, scores, std_err, alpha=0.2)
plt.ylabel('CV score (std error)')
plt.xlabel('alpha')
plt.xticks(np.logspace(-4, 0, 5))
plt.xlim(alphas[0], alphas[-1])
```

Bonus: how much can you trust the selection of alpha?

To answer this question, we use the `LassoCV` object that sets its alpha parameter automatically from the data by internal cross-validation, i.e. it performs cross-validation on the training data it receives.

We use external cross-validation to see how much the automatically obtained alphas differ across different cross-validation folds.

```
lasso_cv = LassoCV(alphas=alphas, cv=5, random_state=0, max_iter=10000)
kf = KFold(3)
print Answer to the bonus question:
    how much can you trust the selection of alpha?
print
print Alpha parameters maximising the generalization score on different
    subsets of the data:
for train, test in enumerate(kf.split(X, y)):
    lasso_cv.fit(X[train], y[train])
    print fold %d, alpha: %f, score: %f % (fold, alphas[lasso_cv.alpha_], lasso_cv.score(X[test], y[test]))
print
print Answer: Not very much, since we obtained different alphas for different
    subsets of the data, and moreover, the scores for these alphas differ.
```

1078 Chapter 5 Examples

scikitlearn user guide Release 0213
printquite substantially
pltshow
Total running time of the script 0 minutes 0294 seconds
514 Feature Selection
Examples concerning the sklearnfeatureselection module
Note Click here to download the full example code
5141 Recursive feature elimination
A recursive feature elimination example showing the relevance of pixels in a digit classification task
Note See also Recursive feature elimination with crossvalidation
514 Feature Selection 1079

```
scikitlearn user guide Release 0213
printdoc
from sklearnsvm import SVC (Support Vector Classification)
from sklearndatasets import loaddigits
from sklearnfeatureselection import RFE
import matplotlib.pyplot as plt
    Load the digits dataset
digits = loaddigits
X = digits.images.reshape(len(digits.images) * 1
y = digits.target
    Create the RFE object and rank each pixel
svc = SVC(kernel='linear', C=1)
rfe = RFE(estimator=svc, n_features_to_select=1, step=1)
rfe.fit(X, y)
ranking = rfe.ranking_.reshape(digits.images[0].shape)
    Plot pixel ranking
plt.matshow(ranking, cmap=plt.cm.Blues)
plt.colorbar
plt.title('Ranking of pixels with RFE')
plt.show
Total running time of the script: 0 minutes 34.36 seconds
Note: Click here to download the full example code
5142 Comparison of Ftest and mutual information
This example illustrates the differences between univariate Ftest statistics and mutual information.
We consider 3 features x1, x2, x3 distributed uniformly over [0, 1]. The target depends on them as follows:
y = x1 * sin(6 * pi * x2) * 0.1 + N(0, 1) that is the third feature is completely irrelevant.
The code below plots the dependency of y against individual xi and normalized values of univariate Ftests statistics
and mutual information.
As Ftest captures only linear dependency, it rates x1 as the most discriminative feature. On the other hand, mutual
information can capture any kind of dependency between variables and it rates x2 as the most discriminative feature,
which probably agrees better with our intuitive perception for this example. Both methods correctly mark x3 as
irrelevant.
```

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.feature_selection import fregression mutualinfo_regression
np.random.seed(0)
X = np.random.rand(1000, 3)
y = X[:, 0] * np.sin(6 * np.pi * X[:, 1] * 0.1) + np.random.randn(1000)
ftest = fregression(X, y)
ftest = np.max(ftest)
mi = mutualinfo_regression(X, y)
mi = np.max(mi)
plt.figure(figsize=(15, 5))
for i in range(3):
    plt.subplot(1, 3, i + 1)
    plt.scatter(X[:, i], y, edgecolor='black', s=20)
    plt.xlabel('x' + str(i + 1), fontsize=14)
    if i == 0:
        plt.ylabel('y', fontsize=14)
    plt.title('Ftest: %f MI: %f' % (ftest, mi),
              fontsize=16)
plt.show
```

Total running time of the script: 0 minutes 00.62 seconds

Note: Click [here](#) to download the full example code

5.14.3 Pipeline: Anova, SVM (Support Vector Machine)

Simple usage of Pipeline that runs successively a univariate feature selection with anova and then a SVM (Support Vector Machine) of the selected features

Using a subpipeline, the fitted coefficients can be mapped back into the original feature space

Out

5.14 Feature Selection 1081

```
scikitlearn user guide Release 0213
precision recall f1score support
0 075 050 060 6
1 067 100 080 6
2 067 080 073 5
3 100 075 086 8
accuracy 076 25
macro avg 077 076 075 25
weighted avg 079 076 076 25
023912131 0 0 0 03236911 0
0 0 0 0 0
010836648 0 0 0 0 0
0 0
043878747 0 0 0 051415652 0
0 0 0 0 0
004845652 0 0 0 0 0
0 0
065382998 0 0 0 057962856 0
0 0 0 0 0
004736524 0 0 0 0 0
0 0
054403412 0 0 0 058478491 0
0 0 0 0 0
011344659 0 0 0 0 0
0 0
from sklearn import svm
from sklearn.datasets import samplesgenerator
from sklearn.feature_selection import SelectKBest fregression
from sklearn.pipeline import makepipeline
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
print doc
import some data to play with
X y samplesgenerator make_classification
nfeatures20 ninformative3 nredundant0 nclasses4
nclustersperclass2
Xtrain Xtest ytrain ytest train_test_split X y randomstate42
ANOVA SVMC
1 anova filter take 3 best ranked features
anovafilter SelectKBest fregression k3
2 svm
clf svmLinearSVC
anova_svm make_pipeline(anovafilter, clf)
anova_svm_fit Xtrain ytrain
ypred anova_svm_predict Xtest
1082 Chapter 5 Examples
```


scikitlearn user guide Release 0213
printclassificationreportytest ypred
coef anovasvm1inversetransformanovasvmlinearsvccoef
printcoef
Total running time of the script 0 minutes 0008 seconds
Note Click here to download the full example code
5144 Recursive feature elimination with crossvalidation
A recursive feature elimination example with automatic tuning of the number of features selected with crossvalidation
Out
Optimal number of features 3
514 Feature Selection 1083

```
scikitlearn user guide Release 0213
printdoc
import matplotlib.pyplot as plt
from sklearnsvm import SVC (Support Vector Classification)
from sklearnmodelselection import StratifiedKFold
from sklearnfeatureselection import RFECV
from sklearndatasets import makeclassification
Build a classification task using 3 informative features
X y makeclassificationnsamples1000 nfeatures25 ninformative3
nredundant2 nrepeated0 nclasses8
nclustersperclass1 randomstate0
Create the RFE object and compute a crossvalidated score
svc SVCkernellinear
The accuracy scoring is proportional to the number of correct
classifications
rfecv RFECVestimatorsvc step1 cvStratifiedKFold2
scoringaccuracy
rfecvfitX y
printOptimal number of features d rfecvnfeatures
Plot number of features VS crossvalidation scores
pltfigure
pltxlabelNumber of features selected
pltylabelCross validation score nb of correct classifications
pltplotrange1 lenrfecvgridscores 1 rfecvgridscores
pltshow
Total running time of the script 0 minutes 1806 seconds
Note Click here to download the full example code
5145 Feature selection using SelectFromModel and LassoCV
Use SelectFromModel metatransformer along with Lasso to select the best couple of features from the Boston dataset
1084 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
Author Manoj Kumar mks542nyuedu
License BSD (Binary Search Decision) 3 clause
printdoc
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_boston
from sklearn.feature_selection import SelectFromModel
from sklearn.linear_model import LassoCV

# Load the boston dataset
boston = load_boston()
X, y = boston.data, boston.target

# We use the base estimator LassoCV since the L1 norm promotes sparsity of features
clf = LassoCV(cv=5)

# Set a minimum threshold of 0.25
sfm = SelectFromModel(clf, threshold=0.25)
sfm.fit(X, y)
n_features_ = sfm.transform(X).shape[1]

# Reset the threshold till the number of features equals two
# Note that the attribute can be set directly instead of repeatedly
514 Feature Selection 1085
```

scikitlearn user guide Release 0213

```
fitting the metatransformer
while nfeatures > 2
    sfmthreshold = 0.1
    Xtransform = sfmtransform(X,
    nfeatures, Xtransform.shape[1])
    Plot the selected two features from X
    plt.title('Features selected from Boston using SelectFromModel with
    threshold = 0.3')
    feature1 = Xtransform[:, 0]
    feature2 = Xtransform[:, 1]
    plt.plot(feature1, feature2, 'r')
    plt.xlabel('Feature number 1')
    plt.ylabel('Feature number 2')
    plt.ylim(np.min(feature2), np.max(feature2))
    plt.show()

Total running time of the script: 0 minutes 00.56 seconds
Note: Click here to download the full example code
```

5146 Test with permutations the significance of a classification score

In order to test if a classification score is significant a technique in repeating the classification procedure after randomizing permuting the labels. The p-value is then given by the percentage of runs for which the score obtained is greater than the classification score obtained in the first place.

1086 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Out
Classification score 05133333333333333333 pvalue 0009900990099009901
Author Alexandre Gramfort alexandregamfortinriafr
License BSD (Binary Search Decision) 3 clause
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearnsvm import SVC (Support Vector Classification)
from sklearnmodelselection import StratifiedKFold
from sklearnmodelselection import permutationtestscore
from sklearn import datasets

Loading a dataset
iris datasetsloadiris
514 Feature Selection 1087
```

scikitlearn user guide Release 0213

```
X irisdata
y iristarget
nclasses npuniqueysize
Some noisy data not correlated
random nprandomRandomStateseed0
E randomnormalsizelenX 2200
Add noisy data to the informative features for make the task harder
X npcX E
svm SVCkernellinear
cv StratifiedKFold2
score permutationscores pvalue permutationtestscore
svm X y scoringaccuracy cvcv npermutations100 njobs1
printClassification score spvalue s score pvalue
```

```
View histogram of permutation scores
plthistpermutationscores 20 labelPermutation scores
edgecolorblack
ylim pltylim
BUG vlins linestyle fails on older versions of matplotlib
pltvlinesscore ylim0 ylim1 linestyle
colorg linewidth3 labelClassification Score
pvalue s pvalue
pltvlines10 nclasses ylim0 ylim1 linestyle
colork linewidth3 labelLuck
pltplot2 score ylim g linewidth3
labelClassification Score
pvalue s pvalue
pltplot2 1 nclasses ylim k linewidth3 labelLuck
pltylimylim
pltlegend
pltxlabelScore
pltshow
```

Total running time of the script 0 minutes 7883 seconds

Note [Click here to download the full example code](#)

5147 Univariate Feature Selection

An example showing univariate feature selection

Noisy non informative features are added to the iris data and univariate feature selection is applied For each feature we plot the pvalues for the univariate feature selection and the corresponding weights of an SVM (Support Vector Machine) We can see that univariate feature selection selects the informative features and that these have larger SVM (Support Vector Machine) weights In the total set of features only the 4 first ones are significant We can see that they have the highest score with univariate feature selection The SVM (Support Vector Machine) assigns a large weight to one of these features but also Selects many of the noninformative features Applying univariate feature selection before the SVM (Support Vector Machine) increases the SVM (Support Vector Machine)

1088 Chapter 5 Examples

```
scikitlearn user guide Release 0213
to the significant features and will thus improve classification
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets svm
from sklearnfeatureselection import SelectPercentile fclassif
```

```
Import some data to play with
The iris dataset
iris datasetsloadiris
Some noisy data not correlated
E np.random.uniform(0, 0.1, size=len(iris.data) * 20)
Add the noisy data to the informative features
X np.hstack((iris.data, E))
y iris.target
plt.figure(1)
plt.clf
514 Feature Selection 1089
```

scikitlearn user guide Release 0213
Xindices np.arange(Xshape1

Univariate feature selection with Ftest for feature scoring
We use the default selection function the 10 most significant features
selector SelectPercentilefclassif percentile10
selectorfitX y
scores nplog10selectorpvalues
scores scoresmax
pltbarXindices 45 scores width2
labelrUnivariate score Logpvalue colordarkorange
edgecolorblack

Compare to the weights of an SVM (Support Vector Machine)
clf svmSVCkernellinear
clffitX y
svmweights clfcoef 2sumaxis0
svmweights svmweightsmax
pltbarXindices 25 svmweights width2 labelSVM weight
colornavy edgecolorblack
clfselected svmSVCkernellinear
clfselectedfitselectortransformX y
svmweightssselected clfselectedcoef 2sumaxis0
svmweightssselected svmweightssselectedmax
pltbarXindicesselectorgetsupport 05 svmweightssselected
width2 labelSVM weights after selection colorc
edgecolorblack
plttitleComparing feature selection
pltxlabelFeature number
plt.xticks
plt.axistight
plt.legendlocupper right
plt.show

Total running time of the script 0 minutes 0045 seconds
515 Gaussian Process for Machine Learning
Examples concerning the sklearngaussianprocess module
Note Click here to download the full example code
1090 Chapter 5 Examples

scikitlearn user guide Release 0213

5151 Illustration of Gaussian process classification GPC on the XOR dataset

This example illustrates GPC on XOR data Compared are a stationary isotropic kernel RBF (Radial Basis Function) and a nonstationary kernel DotProduct On this particular dataset the DotProduct kernel obtains considerably better results because the classboundaries are linear and coincide with the coordinate axes In general stationary kernels often obtain better results

printdoc

Authors Jan Hendrik Metzen jhminformatikunibremende

License BSD (Binary Search Decision) 3 clause

import numpy as np

import matplotlib.pyplot as plt

from sklearn.gaussianprocess import GaussianProcessClassifier

from sklearn.gaussianprocess.kernels import RBF (Radial Basis Function) DotProduct

xx yy np.meshgrid(np.linspace(3, 3, 50)

np.linspace(3, 3, 50)

rng np.random.RandomState(0)

X rng.randn(200, 2)

Y np.logical_xor(X[:, 0] < 0, X[:, 1] < 0)

fit the model

plt.figure(figsize=(10, 5))

kernels = [RBF(lengthscale=10, 10), DotProduct(sigma=0.1)]

for i, kernel in enumerate(kernels):

clf = GaussianProcessClassifier(kernel=kernel, warm_start=True)

plot the decision function for each datapoint on the grid

Z = clf.predict_proba(np.vstack((xx.ravel(), yy.ravel()).T))

Z = Z.reshape(xx.shape)

515 Gaussian Process for Machine Learning 1091

```
scikitlearn user guide Release 0213
pltsubplot1 2 i 1
image pltimshowZ interpolationnearest
extentxxmin xxmax yymin ymax
aspectauto originlower cmappltcmPuOrr
contours pltcontourxx yy Z levels05 linewidths2
colorsk
pltscatterX 0 X 1 s30 cY cmappltcmPaired
edgecolors0 0 0
pltxticks
pltyticks
pltaxis3 3 3 3
pltcolorbarimage
plttitle snLogMarginalLikelihood 3f
 clfkernelflogmarginallikelihoodclfkernelftheta
fontsize12
plttightlayout
pltshow
Total running time of the script 0 minutes 0686 seconds
Note Click here to download the full example code
5152 Gaussian process classification GPC on iris dataset
This example illustrates the predicted probability of GPC for an isotropic and anisotropic RBF (Radial Basis Function) kernel on a two
dimensional version for the irisdataset The anisotropic RBF (Radial Basis Function) kernel obtains slightly higher logmarginallikelihood
by assigning different lengthscales to the two feature dimensions
printdoc
import numpy as np
1092 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.gaussianprocess import GaussianProcessClassifier
from sklearn.gaussianprocess.kernels import RBF (Radial Basis Function)
import some data to play with
iris = datasets.load_iris
X = iris.data[:, 0:2] # we only take the first two features
y = np.array(iris.target)
h = 0.2 # step size in the mesh
kernel = 1.0 * RBF(1.0)
gpcrbfisotropic = GaussianProcessClassifier(kernel=kernel)
kernel = 1.0 * RBF(1.0)
gpcrbfanisotropic = GaussianProcessClassifier(kernel=kernel)
# create a mesh to plot in
xmin, xmax = X[:, 0].min(), X[:, 0].max()
ymin, ymax = X[:, 1].min(), X[:, 1].max()
xx, yy = np.meshgrid(np.arange(xmin, xmax, h),
                     np.arange(ymin, ymax, h))
titles = ('Isotropic RBF (Radial Basis Function)', 'Anisotropic RBF (Radial Basis Function)')
plt.figure(figsize=(10, 5))
for i, clf in enumerate([gpcrbfisotropic, gpcrbfanisotropic]):
    # Plot the predicted probabilities. For that we will assign a color to
    # each point in the mesh
    Z = clf.predict_proba(np.c_[xx.ravel(), yy.ravel()])
    # Put the result into a color plot
    Z = Z.reshape(xx.shape)
    plt.imshow(Z, extent=(xmin, xmax, ymin, ymax), origin='lower')
    # Plot also the training points
    plt.scatter(X[:, 0], X[:, 1], c=y)
    edgecolors = 'r'
    plt.xlabel('Sepal length')
    plt.ylabel('Sepal width')
    plt.xlim(xmin, xmax)
    plt.ylim(ymin, ymax)
    plt.xticks(
    plt.yticks(
    plt.title('LML3f')
    titles[i]
    plt.tight_layout()
plt.show()
Total running time of the script: 0 minutes 42.65 seconds
Note: Click here to download the full example code
515 Gaussian Process for Machine Learning 1093
```

5153 Comparison of kernel ridge and Gaussian process regression

Both kernel ridge regression KRR and Gaussian process regression GPR learn a target function by employing internally the “kernel trick” KRR learns a linear function in the space induced by the respective kernel which corresponds to a nonlinear function in the original space The linear function in the kernel space is chosen based on the meansquared error loss with ridge regularization GPR uses the kernel to define the covariance of a prior distribution over the target functions and uses the observed training data to define a likelihood function Based on Bayes theorem a Gaussian posterior distribution over target functions is defined whose mean is used for prediction A major difference is that GPR can choose the kernel’s hyperparameters based on gradientascent on the marginal likelihood function while KRR needs to perform a grid search on a crossvalidated loss function meansquared error loss A further difference is that GPR learns a generative probabilistic model of the target function and can thus provide meaningful confidence intervals and posterior samples along with the predictions while KRR only provides predictions

This example illustrates both methods on an artificial dataset which consists of a sinusoidal target function and strong noise The figure compares the learned model of KRR and GPR based on a ExpSineSquared kernel which is suited for learning periodic functions The kernel’s hyperparameters control the smoothness l and periodicity of the kernel p Moreover the noise level of the data is learned explicitly by GPR by an additional WhiteKernel component in the kernel and by the regularization parameter alpha of KRR

The figure shows that both methods learn reasonable models of the target function GPR correctly identifies the periodicity of the function to be roughly 2pi 628 while KRR chooses the doubled periodicity 4pi Besides that GPR provides reasonable confidence bounds on the prediction which are not available for KRR A major difference between the two methods is the time required for fitting and predicting while fitting KRR is fast in principle the gridsearch for hyperparameter optimization scales exponentially with the number of hyperparameters “curse of dimensional ity” The gradientbased optimization of the parameters in GPR does not suffer from this exponential scaling and is thus considerable faster on this example with 3dimensional hyperparameter space The time for predicting is similar however generating the variance of the predictive distribution of GPR takes considerable longer than just predicting the mean

Out

Time for KRR fitting 3180
Time for GPR fitting 0096
Time for KRR prediction 0009
1094 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Time for GPR prediction 0010
Time for GPR prediction with standarddeviation 0014
printdoc
  Authors Jan Hendrik Metzen jhminformatikunibremende
  License BSD (Binary Search Decision) 3 clause
import time
import numpy as np
import matplotlib.pyplot as plt
from sklearn.kernelridge import KernelRidge
from sklearn.model_selection import GridSearchCV
from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.gaussian_process.kernels import WhiteKernel, ExpSineSquared
rng = np.random.RandomState(0)
  Generate sample data
X = 15 * rng.rand(100, 1)
y = np.sin(X) * rng.rand(X.shape[0]) + 0.1 * rng.randn(X.shape[0])
  Fit KernelRidge with parameter selection based on 5fold cross validation
paramgrid = {'alpha': [1e0, 1e1, 1e2, 1e3],
              'kernel': [ExpSineSquared, p],
              'forlinn': [2, 2, 10],
              'forpinn': [2, 2, 10]}
kr = GridSearchCV(KernelRidge, paramgrid, cv=5)
stime = time.time()
kr.fit(X, y)
print('Time for KRR fitting: %f' % (time.time() - stime))
gpkernel = ExpSineSquared(10, 50, periodicity_bounds=[1e-2, 1e1],
                           white_kernel=1e1)
gpr = GaussianProcessRegressor(kernel=gpkernel)
stime = time.time()
gpr.fit(X, y)
print('Time for GPR fitting: %f' % (time.time() - stime))
  Predict using kernel ridge
Xplot = np.linspace(0, 20, 10000)
stime = time.time()
ykr = kr.predict(Xplot)
print('Time for KRR prediction: %f' % (time.time() - stime))
  Predict using gaussian process regressor
stime = time.time()
ygpr = gpr.predict(Xplot, return_std=False)
515 Gaussian Process for Machine Learning 1095
```

```
scikitlearn user guide Release 0213
printTime for GPR prediction 3f timetime stime
stime timetime
ygpr ystd gprpredictXplot returnstdTrue
printTime for GPR prediction with standarddeviation 3f
timetime stime
Plot results
pltfigurefigsize10 5
lw 2
pltscatterX y ck labeldata
pltplotXplot npsinXplot colornavy lwlw labelTrue
pltplotXplot ykr colorturquoise lwlw
labelKRR s krbestparams
pltplotXplot ygpr colordarkorange lwlw
labelGPR s gprkernel
pltfillbetweenXplot 0 ygpr ystd ygpr ystd colordarkorange
alpha02
pltxlabeldata
pltylabeltarget
pltxlim0 20
pltylim4 4
plttitleGPR versus Kernel Ridge
pltlegendlocbest scatterpoints1 propsize 8
pltshow
Total running time of the script 0 minutes 3377 seconds
Note Click here to download the full example code
5154 Illustration of prior and posterior Gaussian process for different kernels
This example illustrates the prior and posterior of a GPR with different kernels Mean standard deviation and 10
samples are shown for both prior and posterior
1096 Chapter 5 Examples
```

scikitlearn user guide Release 0213

•

515 Gaussian Process for Machine Learning 1097

scikitlearn user guide Release 0213

•

515 Gaussian Process for Machine Learning 1099

scikitlearn user guide Release 0213

•
printdoc
Authors Jan Hendrik Metzen jhminformatikunibremende

License BSD (Binary Search Decision) 3 clause
import numpy as np
from matplotlib import pyplotasplt
from sklearngaussianprocess import GaussianProcessRegressor
from sklearngaussianprocesskernels import RBF (Radial Basis Function) Matern RationalQuadratic
ExpSineSquared DotProduct
ConstantKernel
kernels 10 RBFlengthscale10 lengthscalebounds1e1 100
515 Gaussian Process for Machine Learning 1101

```
scikitlearn user guide Release 0213
10RationalQuadraticlengthscale10 alpha01
10ExpSineSquaredlengthscale10 periodicity30
lengthscalebounds01 100
periodicitybounds10 100
ConstantKernel01 001 100
DotProductsigma010 sigma0bounds01 100 2
10Maternlengthscale10 lengthscalebounds1e1 100
nu15
forkernelinkernels
Specify Gaussian Process
gp GaussianProcessRegressorkernelkernel
Plot prior
pltfigurefigsize8 8
pltsubplot2 1 1
X nplinspace0 5 100
ymean ystd gppredictX npnewaxis returnstdTrue
pltplotX ymean k lw3 zorder9
pltfillbetweenX ymean ystd ymean ystd
alpha02 colork
ysamples gpsampleyX npnewaxis 10
pltplotX ysamples lw1
pltxlim0 5
pltylim3 3
plttitlePrior kernel s kernel fontsize12
Generate data and fit GP
rng nprandomRandomState4
X rnguniform0 5 10 npnewaxis
y npsinX 0 25 2
gpfitX y
Plot posterior
pltsubplot2 1 2
X nplinspace0 5 100
ymean ystd gppredictX npnewaxis returnstdTrue
pltplotX ymean k lw3 zorder9
pltfillbetweenX ymean ystd ymean ystd
alpha02 colork
ysamples gpsampleyX npnewaxis 10
pltplotX ysamples lw1
pltscatterX 0 y cr s50 zorder10 edgecolors0 0 0
pltxlim0 5
pltylim3 3
plttitlePosterior kernel snLogLikelihood 3f
gpkernel gplogmarginallikelihoodgpkerneltheta
fontsize12
plttightlayout
pltshow
Total running time of the script 0 minutes 1458 seconds
Note Click here to download the full example code
1102 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
5155 Isoprobability lines for Gaussian Processes classification GPC
A twodimensional classification example showing isoprobability lines for the predicted probabilities
Out
Learned kernel 00256 2DotProductsigma0572 2
printdoc
  Author Vincent Dubourg vincentdubourggmailcom
  Adapted to GaussianProcessClassifier
  Jan Hendrik Metzen jhminformatikunibremende
  License BSD (Binary Search Decision) 3 clause
import numpy as np
from matplotlib import pyplotasplt
from matplotlib import cm
from sklearnrgaussianprocess import GaussianProcessClassifier
515 Gaussian Process for Machine Learning 1103
```

```
scikitlearn user guide Release 0213
from sklearn.gaussianprocess.kernels import DotProduct ConstantKernel as C
A few constants
lim = 8
def gx
The function to predict classification will then consist in predicting
whether gx = 0 or not
return 5 * x[1] - 5 * x[0] + 2
Design of experiments
X = np.array([461611719, 600099547,
410469096, 532782448,
000000000, 050000000,
617289014, 46984743,
13109306, 693271427,
503823144, 310584743,
287600388, 674310541,
521301203, 426386883])
Observations
y = np.array(gx, dtype=int)
Instantiate and fit Gaussian Process Model
kernel = C(0.1) * DotProduct(sigma=0.01)
gp = GaussianProcessClassifier(kernel=kernel)
gp.fit(X, y)
print(Learned kernel)
Evaluate real function and the predicted probability
res = 50
x1, x2 = np.meshgrid(np.linspace(lim, lim, res),
np.linspace(lim, lim, res))
xx = np.vstack((x1.reshape(x1.size), x2.reshape(x2.size)))
ytrue = gx
yprob = gp.predict_proba(xx)
ytrue = ytrue.reshape(res)
yprob = yprob.reshape(res)
Plot the probabilistic classification isovalues
fig = plt.figure(1)
ax = fig.gca()
ax.set_aspect('equal')
plt.xticks()
plt.yticks()
ax.set_xticklabels()
ax.set_yticklabels()
plt.xlabel(1)
plt.ylabel(2)
cax = plt.imshow(yprob, cmap=cm.gray, alpha=0.8)
extent = (lim, lim, lim, lim)
norm = plt.matplotlib.colors.Normalize(vmin=0, vmax=0.9)
cb = plt.colorbar(cax, ticks=[0.2, 0.4, 0.6, 0.8, 1], norm=norm)
cb.set_label(r'mathbb{P}(\widehat{G} \leq 0) \text{right}')
plt.clim(0, 1)
```

```
scikitlearn user guide Release 0213
pltplotXy 0 0 Xy 0 1 r markersize12
pltplotXy 0 0 Xy 0 1 b markersize12
pltcontourx1 x2 ytrue 0 colorsk linestyledashdot
cs pltcontourx1 x2 yprob 0666 colorsb
linestylelessolid
pltlabelcs fontsize11
cs pltcontourx1 x2 yprob 05 colorsk
linestylesdashed
pltlabelcs fontsize11
cs pltcontourx1 x2 yprob 0334 colorsr
linestylelessolid
pltlabelcs fontsize11
pltshow
```

Total running time of the script 0 minutes 0098 seconds

Note [Click here to download the full example code](#)

5156 Probabilistic predictions with Gaussian process classification GPC

This example illustrates the predicted probability of GPC for an RBF (Radial Basis Function) kernel with different choices of the hyperparameters The first figure shows the predicted probability of GPC with arbitrarily chosen hyperparameters and with the hyperparameters corresponding to the maximum logmarginallikelihood LML

While the hyperparameters chosen by optimizing LML have a considerable larger LML they perform slightly worse according to the logloss on test data The figure shows that this is because they exhibit a steep change of the class probabilities at the class boundaries which is good but have predicted probabilities close to 05 far away from the class boundaries which is bad This undesirable effect is caused by the Laplace approximation used internally by GPC

The second figure shows the logmarginallikelihood for different choices of the kernel’s hyperparameters highlighting the two choices of the hyperparameters used in the first figure by black dots

scikitlearn user guide Release 0213

•
Out
Log Marginal Likelihood initial 17598
Log Marginal Likelihood optimized 3875
Accuracy 1000 initial 1000 optimized
Logloss 0214 initial 0319 optimized
printdoc
Authors Jan Hendrik Metzen jhminformatikunibremende

License BSD (Binary Search Decision) 3 clause
import numpy as np
from matplotlib import pyplot as plt
from sklearn.metrics.classification import accuracy_score, log_loss
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF (Radial Basis Function)
515 Gaussian Process for Machine Learning 1107

```
scikitlearn user guide Release 0213
Generate data
trainsize 50
rng nprandomRandomState0
X rnguniform0 5 100 npnewaxis
y nparrayX 0 25 dtypeint
Specify Gaussian Processes with fixed and optimized hyperparameters
gpfix GaussianProcessClassifierkernel10 RBFlengthscale10
optimizerNone
gpfixfitXtrainsize ytrainsize
gpopt GaussianProcessClassifierkernel10 RBFlengthscale10
gpoptfitXtrainsize ytrainsize
printLog Marginal Likelihood initial 3f
gpfixlogmarginallikelihoodgpfixkerneltheta
printLog Marginal Likelihood optimized 3f
gpoptlogmarginallikelihoodgpoptkerneltheta
printAccuracy 3finitial 3foptimized
accuracycoreytrainsize gpfixpredictXtrainsize
accuracycoreytrainsize gpoptpredictXtrainsize
printLogloss 3finitial 3foptimized
loglossytrainsize gpfixpredictprobaXtrainsize 1
loglossytrainsize gpoptpredictprobaXtrainsize 1
Plot posteriors
pltfigure
pltscatterXtrainsize 0 ytrainsize ck labelTrain data
edgecolors0 0 0
pltscatterXtrainsize 0 ytrainsize cg labelTest data
edgecolors0 0 0
X nplinspace0 5 100
pltplotX gpfixpredictprobaX npnewaxis 1 r
labelInitial kernel s gpfixkernel
pltplotX gpoptpredictprobaX npnewaxis 1 b
labelOptimized kernel s gpoptkernel
pltxlabelFeature
pltylabelClass 1 probability
pltxlim0 5
pltylim025 15
pltlegendlocbest
Plot LML landscape
pltfigure
theta0 nplogspace0 8 30
theta1 nplogspace1 1 29
Theta0 Theta1 npmeshgridtheta0 theta1
LML gpoptlogmarginallikelihoodnplogTheta0i j Theta1i j
foriinrangeTheta0shape0 forjinrangeTheta0shape1
LML nparrayLMLT
pltplotnpexpgpfixkerneltheta0 npexpgpfixkerneltheta1
ko zorder10
pltplotnpexpgpoptkerneltheta0 npexpgpoptkerneltheta1
ko zorder10
pltptcolorTheta0 Theta1 LML
pltyscalelog
1108 Chapter 5 Examples
```

scikitlearn user guide Release 0213

pltyscalelog
pltcolorbar
pltxlabelMagnitude
pltylabelLengthscale
plttitleLogmarginallikelihood
pltshow

Total running time of the script 0 minutes 2514 seconds

Note Click here to download the full example code

5157 Gaussian process regression GPR with noiselevel estimation

This example illustrates that GPR with a sumkernel including a WhiteKernel can estimate the noise level of data
An illustration of the logmarginallikelihood LML landscape shows that there exist two local maxima of LML The
first corresponds to a model with a high noise level and a large length scale which explains all variations in the data
by noise The second one has a smaller noise level and shorter length scale which explains most of the variation by
the noisefree functional relationship The second model has a higher likelihood however depending on the initial
value for the hyperparameters the gradientbased optimization might also converge to the highnoise solution It is
thus important to repeat the optimization several times for different initializations

-

515 Gaussian Process for Machine Learning 1109

scikitlearn user guide Release 0213

•
printdoc
Authors Jan Hendrik Metzen jhminformatikunibremende

License BSD (Binary Search Decision) 3 clause
import numpy as np
from matplotlib import pyplotasplt
from matplotlibcolors import LogNorm
from sklearngaussianprocess import GaussianProcessRegressor
from sklearngaussianprocesskernels import RBF (Radial Basis Function) WhiteKernel
rng np.random.RandomState0
X rng.uniform0 5 20 np.newaxis
y 0.5 np.sin(3 * X) rng.normal0 0.5 X.shape0
First run
plt.figure
kernel 10 RBF.lengthscale1000 lengthscale.bounds1e2 1e3
WhiteKernel.noiselevel1 noiselevel.bounds1e10 1e1
gp GaussianProcessRegressor(kernel=kernel
alpha=0.0).fit(X, y)
X np.linspace0 5 100
y.mean(), y.cov(), gp.predict(X, np.newaxis, return_cov=True)
515 Gaussian Process for Machine Learning 1111

```
scikitlearn user guide Release 0213
pltplotX ymean k lw3 zorder9
pltfillbetweenX ymean npsqrtnpdiagetcov
ymean npsqrtnpdiagetcov
alpha05 colork
pltplotX 05 npsin3 X r lw3 zorder9
pltscatterX 0 y cr s50 zorder10 edgecolors0 0 0
plttitleInitial snOptimum snLogMarginalLikelihood s
kernel gpkernel
gplogmarginallikelihoodgpkerneltheta
plttightlayout
Second run
pltfigure
kernel 10 RBFlengthiscale10 lengthscalebounds1e2 1e3
WhiteKernelnoiselevel1e5 noiselevelbounds1e10 1e1
gp GaussianProcessRegressorkernelkernel
alpha00fitX y
X nplinspace0 5 100
ymean ycov gppredictX npnewaxis returncovTrue
pltplotX ymean k lw3 zorder9
pltfillbetweenX ymean npsqrtnpdiagetcov
ymean npsqrtnpdiagetcov
alpha05 colork
pltplotX 05 npsin3 X r lw3 zorder9
pltscatterX 0 y cr s50 zorder10 edgecolors0 0 0
plttitleInitial snOptimum snLogMarginalLikelihood s
kernel gpkernel
gplogmarginallikelihoodgpkerneltheta
plttightlayout
Plot LML landscape
pltfigure
theta0 nplogspace2 3 49
theta1 nplogspace2 0 50
Theta0 Theta1 npmeshgridtheta0 theta1
LML gplogmarginallikelihoodnplog036 Theta0i j Theta1i j
foriinrangeTheta0shape0 forjinrangeTheta0shape1
LML nparrayLMLT
vmin vmax LMLmin LMLmax
vmax 50
level nparoundnplogspacenplog10vmin nplog10vmax 50 decimals1
pltcontourTheta0 Theta1 LML
levelslevel normLogNormvminvmin vmaxvmax
pltcolorbar
pltyscalelog
pltyscalelog
pltxlabelLengthscale
pltylabelNoiselevel
plttitleLogmarginallikelihood
plttightlayout
pltshow
Total running time of the script 0 minutes 2874 seconds
1112 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Note Click here to download the full example code

5158 Gaussian Processes regression basic introductory example

A simple onedimensional regression example computed in two different ways

1 A noisefree case

2 A noisy case with known noiselevel per datapoint

In both cases the kernel's parameters are estimated using the maximum likelihood principle

The figures illustrate the interpolating property of the Gaussian Process model as well as its probabilistic nature in the form of a pointwise 95 confidence interval

Note that the parameter alpha is applied as a Tikhonov regularization of the assumed covariance between the training points

•

515 Gaussian Process for Machine Learning 1113

scikitlearn user guide Release 0213

```
•
printdoc
Author Vincent Dubourg vincentdubourggmailcom
Jake Vanderplas vanderplasastronwashingtonedu
Jan Hendrik Metzen jhminformatikunibremendes
License BSD (Binary Search Decision) 3 clause
import numpy as np
from matplotlib import pyplot as plt
from sklearn.gaussianprocess import GaussianProcessRegressor
from sklearn.gaussianprocess.kernels import RBF (Radial Basis Function) ConstantKernel as C
np.random.seed(1)
def f(x):
    The function to predict
    return x * np.sin(x)

# First the noiseless case
X = np.linspace(0, 1, 100)
y = f(X)

1114 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
Mesh the input space for evaluations of the real function the prediction and
its MSE
x = np.linspace(0, 10, 1000)
Instantiate a Gaussian Process model
kernel = C10(1e3, 1e3, RBF10(1e2, 1e2))
gp = GaussianProcessRegressor(kernel=kernel, n_restart_optimizer=9)
Fit to data using Maximum Likelihood Estimation of the parameters
gp.fit(X, y)
Make the prediction on the meshed x-axis ask for MSE as well
ypred, sigma = gp.predict(x, return_std=True)
Plot the function the prediction and the 95 confidence interval based on
the MSE
plt.figure()
plt.plot(x, f(x), label='f(x)')
plt.plot(x, y, 'r', markersize=10, label='Observations')
plt.plot(x, ypred, 'b', label='Prediction')
plt.fill(np.concatenate(x, x1),
np.concatenate(ypred, 19600 * sigma),
ypred, 19600 * sigma)
alpha = 0.5
fig, ax = plt.subplots()
ax.plot(x, f(x), label='f(x)')
ax.plot(x, y, 'r', markersize=10, label='Observations')
ax.plot(x, ypred, 'b', label='Prediction')
ax.fill(np.concatenate(x, x1),
np.concatenate(ypred, 19600 * sigma),
ypred, 19600 * sigma)
ax.legend(loc='upper left')

# now the noisy case
X = np.linspace(0, 1, 99)
X = np.linspace(0, 1, 99)
Observations and noise
y = f(X) + noise
dy = 0.05
noise = np.random.normal(0, dy)
y = noise
Instantiate a Gaussian Process model
gp = GaussianProcessRegressor(kernel=kernel, alpha=dy**2,
n_restart_optimizer=10)
Fit to data using Maximum Likelihood Estimation of the parameters
gp.fit(X, y)
Make the prediction on the meshed x-axis ask for MSE as well
ypred, sigma = gp.predict(x, return_std=True)
Plot the function the prediction and the 95 confidence interval based on
the MSE
plt.figure()
plt.plot(x, f(x), label='f(x)')
plt.errorbar(X, y, dy, fmtr=markersize=10, label='Observations')
plt.plot(x, ypred, 'b', label='Prediction')
plt.fill(np.concatenate(x, x1),
np.concatenate(ypred, 19600 * sigma),
ypred, 19600 * sigma)
515 Gaussian Process for Machine Learning 1115
```

scikitlearn user guide Release 0213

ypred 19600 sigma1

alpha5 fcb ecNone label95 confidence interval

pltxlabelx

pltylabelfx

pltylim10 20

pltlegendlocupper left

pltshow

Total running time of the script 0 minutes 0284 seconds

Note Click here to download the full example code

5159 Gaussian process regression GPR on Mauna Loa CO2 data

This example is based on Section 543 of “Gaussian Processes for Machine Learning” RW2006 It illustrates an example of complex kernel engineering and hyperparameter optimization using gradient ascent on the logmarginal likelihood The data consists of the monthly average atmospheric CO2 concentrations in parts per million by volume ppmv collected at the Mauna Loa Observatory in Hawaii between 1958 and 2001 The objective is to model the CO2 concentration as a function of the time t

The kernel is composed of several terms that are responsible for explaining different properties of the signal

- a long term smooth rising trend is to be explained by an RBF (Radial Basis Function) kernel The RBF (Radial Basis Function) kernel with a large lengthscale enforces this component to be smooth it is not enforced that the trend is rising which leaves this choice to the GP The specific lengthscale and the amplitude are free hyperparameters

- a seasonal component which is to be explained by the periodic ExpSineSquared kernel with a fixed periodicity of 1 year The lengthscale of this periodic component controlling its smoothness is a free parameter In order to allow decaying away from exact periodicity the product with an RBF (Radial Basis Function) kernel is taken The lengthscale of this RBF (Radial Basis Function) component controls the decay time and is a further free parameter

- smaller medium term irregularities are to be explained by a RationalQuadratic kernel component whose lengthscale and alpha parameter which determines the diffuseness of the lengthscales are to be determined According to RW2006 these irregularities can better be explained by a RationalQuadratic than an RBF (Radial Basis Function) kernel component probably because it can accommodate several lengthscales

- a “noise” term consisting of an RBF (Radial Basis Function) kernel contribution which shall explain the correlated noise components such as local weather phenomena and a WhiteKernel contribution for the white noise The relative amplitudes and the RBF (Radial Basis Function)’s length scale are further free parameters

Maximizing the logmarginallikelihood after subtracting the target’s mean yields the following kernel with an LML of 83214

3442RBFlengthscale418

3272RBFlengthscale180 ExpSineSquaredlengthscale144 periodicity1

04462RationalQuadraticalpha177 lengthscale0957

01972RBFlengthscale0138 WhiteKernelnoiselevel00336

Thus most of the target signal 344ppm is explained by a longterm rising trend lengthscale 418 years The periodic component has an amplitude of 327ppm a decay time of 180 years and a lengthscale of 144 The long decay time indicates that we have a locally very close to periodic seasonal component The correlated noise has an amplitude of 0197ppm with a length scale of 0138 years and a whitenoise contribution of 0197ppm Thus the 1116 Chapter 5 Examples

scikitlearn user guide Release 0213

overall noise level is very small indicating that the data can be very well explained by the model The figure shows also that the model makes very confident predictions until around 2015

Out

GPML kernel 66 2RBFlengthscale67 24 2RBFlengthscale90

↪ExpSineSquaredlengthscale13 periodicity1 066 2

↪RationalQuadraticalpha078 lengthscale12 018 2RBFlengthscale0134

↪ WhiteKernelnoiselevel00361

Logmarginallikelihood 117023

Learned kernel 448 2RBFlengthscale516 264 2RBFlengthscale915

↪ExpSineSquaredlengthscale148 periodicity1 0536 2

↪RationalQuadraticalpha289 lengthscale0968 0188 2RBFlengthscale0

↪122 WhiteKernelnoiselevel00367

Logmarginallikelihood 115050

Authors Jan Hendrik Metzen jhminformatikunibremende

License BSD (Binary Search Decision) 3 clause

515 Gaussian Process for Machine Learning 1117

```
scikitlearn user guide Release 0213
import numpy as np
from matplotlib import pyplot as plt
from sklearn.datasets import fetch_openml
from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.gaussian_process.kernels import
import RBF, WhiteKernel, RationalQuadratic, ExpSineSquared
print doc
def load_mauna_loa_atmospheric_co2
mldata = fetch_openml(data_id=41187)
months =
ppmv_sums =
counts =
y = mldata.data[:, 0]
m = mldata.data[:, 1]
month_float = y / m * 12
ppmv_s = mldata.target
for month, ppmv in zip(month_float, ppmv_s):
    if not month in month_list:
        month_list.append(month)
        ppmv_sums.append(ppmv)
        counts.append(1)
    else:
        # aggregate monthly sum to produce average
        ppmv_sums[month] += ppmv
        counts[month] += 1
months = np.array(month_list)
avg_ppmv_s = np.array(ppmv_sums) / counts
return months, avg_ppmv_s

X, y = load_mauna_loa_atmospheric_co2
# Kernel with parameters given in GPML book
k1 = 660 * RBF(length_scale=670) # long term smooth rising trend
k2 = 24 * RBF(length_scale=900)
ExpSineSquared(length_scale=13, periodicity=10) # seasonal component
# medium term irregularity
k3 = 0.66 * 2
RationalQuadratic(length_scale=12, alpha=0.78)
k4 = 0.18 * RBF(length_scale=0.134)
# WhiteKernel noise level 0.19, 2 noise terms
kernel_gpml = k1 * k2 * k3 * k4
gp = GaussianProcessRegressor(kernel=kernel_gpml, alpha=0,
optimizer=None, normalize_y=True)
gp.fit(X, y)
print GPML kernel: %s % kernel
print Log marginal likelihood: %f % gp.log_marginal_likelihood(gp.kernel.theta)
1118 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Kernel with optimized parameters

k1 500 2RBFlengthscale500 long term smooth rising trend

k2 20 2RBFlengthscale1000

ExpSineSquaredlengthscale10 periodicity10

periodicityboundsfixed seasonal component

medium term irregularities

k3 05 2RationalQuadraticlengthscale10 alpha10

k4 01 2RBFlengthscale01

WhiteKernelnoiselevel01 2

noiselevelbounds1e3 npinf noise terms

kernel k1 k2 k3 k4

gp GaussianProcessRegressorkernelkernel alpha0

normalizeyTrue

gpfitX y

printnLearned kernel s gpkernel

printLogmarginallikelihood 3f

gplogmarginallikelihoodgpkerneltheta

X np.linspace(Xmin Xmax 30 1000 np.newaxis

ypred ystd gppredictX returnstdTrue

Illustration

plt.scatter(X y ck

plt.plot(X ypred

plt.fill_between(X 0 ypred ystd ypred ystd

alpha05 color k

plt.xlim(Xmin Xmax

plt.ylabel(Year

plt.ylabel(rCO2 in ppm

plt.title(Atmospheric CO2 concentration at Mauna Loa

plt.tight_layout

plt.show

Total running time of the script 0 minutes 11550 seconds

516 Missing Value Imputation

Examples concerning the sklearnimpute module

Note Click here to download the full example code

5161 Imputing missing values with variants of IterativeImputer

The sklearnimpute.IterativeImputer class is very flexible it can be used with a variety of estimators

to do roundrobin regression treating every variable as an output in turn

In this example we compare some estimators for the purpose of missing feature imputation with sklearnimpute

IterativeImputer

- BayesianRidge regularized linear regression

516 Missing Value Imputation 1119

scikitlearn user guide Release 0213

- DecisionTreeRegressor nonlinear regression
- ExtraTreesRegressor similar to missForest in R
- KNeighborsRegressor comparable to other KNN imputation approaches

Of particular interest is the ability of sklearnimputeIterativeImputer to mimic the behavior of missForest a popular imputation package for R In this example we have chosen to use sklearnensembleExtraTreesRegressor instead of sklearnensembleRandomForestRegressor as in missForest due to its increased speed

Note that sklearnneighborsKNeighborsRegressor is different from KNN imputation which learns from samples with missing values by using a distance metric that accounts for missing values rather than imputing them

The goal is to compare different estimators to see which one is best for the sklearnimputeIterativeImputer when using a sklearnlinearmodelBayesianRidge estimator on the California housing dataset with a single value randomly removed from each row

For this particular pattern of missing values we see that sklearnensembleExtraTreesRegressor and sklearnlinearmodelBayesianRidge give the best results

```
printdoc
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
To use this experimental feature we need to explicitly ask for it
from sklearnexperimental import enableiterativeimputer noqa
from sklearndatasets import fetchcaliforniahousing
from sklearnimpute import SimpleImputer
from sklearnimpute import IterativeImputer
from sklearnlinearmodel import BayesianRidge
from sklearntree import DecisionTreeRegressor
from sklearnensemble import ExtraTreesRegressor
from sklearnneighbors import KNeighborsRegressor
from sklearnpipeline import makepipeline
from sklearnmodelselection import crossvalscore
1120 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
NSPLITS 5
rng np.random.RandomState(0)
Xfull yfull fetchcaliforniahousingreturnXy=True
2k samples is enough for the purpose of the example
Remove the following two lines for a slower run with different error bars
Xfull Xfull10
yfull yfull10
nsamples nfeatures Xfull.shape
Estimate the score on the entire dataset with no missing values
brestimator BayesianRidge
scorefulldata pd.DataFrame
crossvalscore
brestimator Xfull yfull scoring=negmeansquarederror
cvNSPLITS
```

columnsFull Data

```
Add a single missing value to each row
Xmissing Xfull.copy()
ymissing yfull
missingsamples np.arange(samples)
missingfeatures rng.choice(nfeatures, nsamples, replace=True)
Xmissing[missingsamples, missingfeatures] = np.nan
Estimate the score after imputation mean and median strategies
scoresimpleimputer pd.DataFrame
forstrategy in ['mean', 'median']:
    estimator = make_pipeline(
        SimpleImputer(missing_values=np.nan, strategy=strategy),
        brestimator
```

```
scoresimpleimputer(strategy) crossvalscore
estimator Xmissing ymissing scoring=negmeansquarederror
cvNSPLITS
```

```
Estimate the score after iterative imputation of the missing values
with different estimators
estimators = [
    BayesianRidge,
    DecisionTreeRegressor(max_features=sqrt, random_state=0),
    ExtraTreesRegressor(n_estimators=10, random_state=0),
    KNeighborsRegressor(n_neighbors=15)
```

```
scoreiterativeimputer pd.DataFrame
forimputeestimator in estimators:
    estimator = make_pipeline(
        IterativeImputer(random_state=0), estimator, imputeestimator
    )
    brestimator
```

```
scoreiterativeimputer(imputeestimator, classname)
crossvalscore
estimator Xmissing ymissing scoring=negmeansquarederror
cvNSPLITS
```

scikitlearn user guide Release 0213

```
scores = pdconcat
scorefulldata scoresimpleimputer scoreiterativeimputer
keysOriginal SimpleImputer IterativeImputer axis1
```

```
plot boston results
fig ax = pltsubplots(figsize=(13, 6))
means = scores.mean
errors = scores.std
meansplotbarhxerrerrors axax
ax.set_title('California Housing Regression with Different Imputation Methods')
ax.set_xlabel('MSE, smaller is better')
ax.set_yticks(np.arange(means.shape[0]))
ax.set_yticklabels(w_join(label_for_label_in_means_index_get_values))
plt.tight_layout(pad=1)
plt.show
```

Total running time of the script: 0 minutes 19.017 seconds

Note: Click [here](#) to download the full example code

5162 Imputing missing values before building an estimator

Missing values can be replaced by the mean, the median, or the most frequent value using the basic sklearn `impute.SimpleImputer`. The median is a more robust estimator for data with high magnitude variables which could dominate results otherwise known as a ‘long tail’.

Another option is the sklearn `impute.IterativeImputer`. This uses round-robin linear regression treating every variable as an output in turn. The version implemented assumes Gaussian output variables. If your features are obviously non-normal, consider transforming them to look more normal so as to potentially improve performance. In addition to using an imputing method, we can also keep an indication of the missing information using sklearn `impute.MissingIndicator` which might carry some information.

1122 Chapter 5 Examples


```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
    To use the experimental IterativeImputer we need to explicitly ask for it
from sklearn.experimental import enableiterativeimputer noqa
from sklearn.datasets import load_diabetes
from sklearn.datasets import load_boston
from sklearn.ensemble import RandomForestRegressor
from sklearn.pipeline import make_pipeline, make_union
from sklearn.impute import SimpleImputer, IterativeImputer, MissingIndicator
from sklearn.model_selection import cross_val_score
rng = np.random.RandomState(0)
NSPLITS = 5
REGRESSOR = RandomForestRegressor(random_state=0, n_estimators=100)
def get_scores_for_imputer(imputer, X_missing, y_missing):
    estimator = make_pipeline(
        make_union(imputer, MissingIndicator(missing_values=0)),
        REGRESSOR
    )
    impute_scores = cross_val_score(estimator, X_missing, y_missing,
        scoring='neg_mean_squared_error',
        cv=NSPLITS,
        return_impute_scores=True)
    def get_results_dataset(
        X_full, y_full, dataset_data, dataset_target):
        n_samples = X_full.shape[0]
        n_features = X_full.shape[1]
        Estimate the score on the entire dataset with no missing values
    516 Missing Value Imputation 1123
```

scikitlearn user guide Release 0213
fullscores crossvalscoreREGRESSOR Xfull yfull
scoringnegmeansquarederror
cvNSPLITS
Add missing values in 75 of the lines
missingrate 0.75
nmissingsamples intnpfloornsamples missingrate
missingsamples nphstacknpzerosnsamples nmissingsamples
dtype npbool
np.ones(nmissingsamples)
dtype npbool
rng.shuffle(missingsamples)
missingfeatures rng.randint(0, nfeatures, nmissingsamples)
Xmissing Xfullcopy
Xmissing[np.where(missingsamples)] missingfeatures 0
ymissing yfullcopy
Estimate the score after replacing missing values by 0
imputer SimpleImputer(missing_values=0, strategy='constant',
fill_value=0)
zero_impute_scores get_scores_for_imputer(imputer, Xmissing, ymissing)
Estimate the score after imputation mean strategy of the missing values
imputer SimpleImputer(missing_values=0, strategy='mean',
mean_impute_scores get_scores_for_imputer(imputer, Xmissing, ymissing)
Estimate the score after iterative imputation of the missing values
imputer IterativeImputer(missing_values=0, random_state=0,
nearest_features=5)
iterative_impute_scores get_scores_for_imputer(imputer, Xmissing, ymissing)
return_full_scores_mean full_scores_std
zero_impute_scores_mean zero_impute_scores_std
mean_impute_scores_mean mean_impute_scores_std
iterative_impute_scores_mean iterative_impute_scores_std
results_diabetes np.array(get_results_load_diabetes, mses_diabetes, results_diabetes[0], 1, stds_diabetes, results_diabetes[1], results_boston, np.array(get_results_load_boston, mses_boston, results_boston[0], 1, stds_boston, results_boston[1], nbars, len(mses_diabetes), xval, npar, ngen, nbars, xlabels, 'Full data', 'Zero imputation', 'Mean Imputation', 'Multivariate Imputation', colors, 'r g b orange', 1124, 'Chapter 5 Examples')
1124 Chapter 5 Examples

```
scikitlearn user guide Release 0213
plot diabetes results
pltfigurefigsize12 6
ax1 pltsubplot121
forjinxval
ax1barhj msesdiabetesj xerrstdsdiabetesj
colorcolorsj alpha06 aligncenter
ax1settitleImputation Techniques with Diabetes Data
ax1setxlimleftnpminmsesdiabetes 09
rightnpmaxmsesdiabetes 11
ax1setyticksxval
ax1setxlabelMSE
ax1invertaxis
ax1setyticklabelsxlabels
plot boston results
ax2 pltsubplot122
forjinxval
ax2barhj msesbostonj xerrstdsbostonj
colorcolorsj alpha06 aligncenter
ax2settitleImputation Techniques with Boston Data
ax2setyticksxval
ax2setxlabelMSE
ax2invertaxis
ax2setyticklabels nbars
pltshow
Total running time of the script 0 minutes 11569 seconds
517 Inspection
Examples related to the sklearninspection module
Note Click here to download the full example code
5171 Partial Dependence Plots
Partial dependence plots show the dependence between the target function2and a set of ‘target’ features marginalizing
over the values of all other features the complement features Due to the limits of human perception the size of the
target feature set must be small usually one or two thus the target features are usually chosen among the most
important features
This example shows how to obtain partial dependence plots from a MLPRegressor and a
GradientBoostingRegressor trained on the California housing dataset The example is taken from1
The plots show four 1way and two 1way partial dependence plots omitted for MLPRegressor due to computa
tion time The target variables for the oneway PDP are median income MedInc average occupants per household
AvgOccup median house age HouseAge and average rooms per household AveRooms
2For classification you can think of it as the regression score before the link function
1T Hastie R Tibshirani and J Friedman “Elements of Statistical Learning Ed 2” Springer 2009
517 Inspection 1125
```

scikitlearn user guide Release 0213

We can clearly see that the median house price shows a linear relationship with the median income top left and that the house price drops when the average occupants per household increases top middle The top right plot shows that the house age in a district does not have a strong influence on the median house price so does the average rooms per household The tick marks on the xaxis represent the deciles of the feature values in the training data We also observe that MLPRegressor has much smoother predictions than GradientBoostingRegressor For the plots to be comparable it is necessary to subtract the average value of the target y The 'recursion' method used by default for GradientBoostingRegressor does not account for the initial predictor in our case the average target Setting the target average to 0 avoids this bias

Partial dependence plots with two target features enable us to visualize interactions among them The twoway partial dependence plot shows the dependence of median house price on joint values of house age and average occupants per household We can clearly see an interaction between the two features for an average occupancy greater than two the house price is nearly independent of the house age whereas for values less than two there is a strong dependence on age

On a third figure we have plotted the same partial dependence plot this time in 3 dimensions

•

scikitlearn user guide Release 0213

- 517 Inspection 1127

scikitlearn user guide Release 0213

•

Out

Training MLPRegressor

Computing partial dependence plots

Training GradientBoostingRegressor

Computing partial dependence plots

Custom 3d plot via partialdependence

printdoc

import numpy as np

import matplotlib.pyplot as plt

from mpl_toolkits.mplot3d import Axes3D

from sklearn.inspection import partialdependence

from sklearn.inspection import plot_partialdependence

from sklearn.ensemble import GradientBoostingRegressor

from sklearn.neural_network import MLPRegressor

from sklearn.datasets.california_housing import fetch_california_housing

def main

1128 Chapter 5 Examples

```
scikitlearn user guide Release 0213
calhousing fetchcaliforniahousing
X y calhousingdata calhousingtarget
names calhousingfeaturenames
Center target to avoid gradient boosting init bias gradient boosting
with the recursion method does not account for the initial estimator
here the average target by default
y ymean
printTraining MLPRegressor
est MLPRegressoractivationlogistic
estfitX y
printComputing partial dependence plots
We dont compute the 2way PDP 5 1 here because it is a lot slower
with the brute method
features 0 5 1 2
plotpartialdependenceest X features featurenamesnames
njobs3 gridresolution50
fig pltgcf
figsuptitlePartial dependence of house value on nonlocation features n
for the California housing dataset with MLPRegressor
pltsubplotsadjusttop09 tightlayout causes overlap with suptitle
printTraining GradientBoostingRegressor
est GradientBoostingRegressorn_estimators100 maxdepth4
learningrate01 losshuber
randomstate1
estfitX y
printComputing partial dependence plots
features 0 5 1 2 5 1
plotpartialdependenceest X features featurenamesnames
njobs3 gridresolution50
fig pltgcf
figsuptitlePartial dependence of house value on nonlocation features n
for the California housing dataset with Gradient Boosting
pltsubplotsadjusttop09
printCustom 3d plot via partialdependence
fig pltfigure
targetfeature 1 5
pdp axes partialdependenceest X targetfeature
gridresolution50
XX YY npmeshgridaxes0 axes1
Z pdp0T
ax Axes3Dfig
surf axplotsurfaceXX YY Z rstride1 cstride1
cmapppltcmBuPu edgecolork
axsetxlabelnamestargetfeature0
axsetylabelnamestargetfeature1
axsetzlabelPartial dependence
pretty init view
axviewinitelev22 azim122
pltcolorbarsurf
pltsubplotsadjusttop09
517 Inspection 1129
```

scikitlearn user guide Release 0213
pltshow
 Needed on Windows because plotpartialdependence uses multiprocessing
ifname main
main
Total running time of the script 0 minutes 24838 seconds
518 Generalized Linear Models
Examples concerning the sklearnlinearmodel module
Note Click here to download the full example code
5181 Lasso path using LARS
Computes Lasso Path along the regularization parameter using the LARS algorithm on the diabetes dataset Each
color represents a different feature of the coefficient vector and this is displayed as a function of the regularization
parameter
1130 Chapter 5 Examples


```
scikitlearn user guide Release 0213
Out
Computing regularization path using the LARS

printdoc
Author Fabian Pedregosa fabianpedregosainriafr
Alexandre Gramfort alexandregamfortinriafr
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn import datasets
diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target
518 Generalized Linear Models 1131
```

scikitlearn user guide Release 0213

printComputing regularization path using the LARS

coefs linearmodellarspathX y methodlasso verboseTrue

xx npsumnpabscoefsT axis1

xx xx1

pltplotxx coefsT

ymin ymax pltylim

pltvlinesxx ymin ymax linestyledashed

pltxlabelcoef maxcoef

pltylabelCoefficients

plttitleLASSO Path

pltaxistight

pltshow

Total running time of the script 0 minutes 0025 seconds

Note Click here to download the full example code

5182 Plot Ridge coefficients as a function of the regularization

Shows the effect of collinearity in the coefficients of an estimator

Ridge Regression is the estimator used in this example Each color represents a different feature of the coefficient

vector and this is displayed as a function of the regularization parameter

This example also shows the usefulness of applying Ridge regression to highly illconditioned matrices For such

matrices a slight change in the target variable can cause huge variances in the calculated weights In such cases it is

useful to set a certain regularization alpha to reduce this variation noise

When alpha is very large the regularization effect dominates the squared loss function and the coefficients tend to

zero At the end of the path as alpha tends toward zero and the solution tends towards the ordinary least squares

coefficients exhibit big oscillations In practise it is necessary to tune alpha in such a way that a balance is maintained

between both

1132 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Author Fabian Pedregosa fabianpedregosainriafr
License BSD (Binary Search Decision) 3 clause
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
X is the 10x10 Hilbert matrix
X 1 nparange1 11 nparange0 10 npnewaxis
y npones10

Compute paths
nalphas 200
alphas nplogspace10 2 nalphas
coefs
forainalphas
ridge linear_model.Ridge(alpha=aa fit_intercept=False
ridge_fit(X, y)
coefs.append(ridge.coef
```

scikitlearn user guide Release 0213

 Display results

ax = plt.gca

ax.plotalphas_coefs

ax.setxscalelog

ax.setxlimaxgetxlim1 reverse axis

plt.xlabelalpha

plt.ylabelweights

plt.titleRidge coefficients as a function of the regularization

plt.axis('tight')

plt.show

Total running time of the script: 0 minutes 0161 seconds

Note: Click [here](#) to download the full example code

5183 SGD Maximum margin separating hyperplane

Plot the maximum margin separating hyperplane within a twoclass separable dataset using a linear Support Vector

Machines classifier trained using SGD

1134 Chapter 5 Examples

scikitlearn user guide Release 0213

```
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import SGDClassifier
from sklearn.datasets.samples_generator import make_blobs
# we create 50 separable points
X, Y = make_blobs(n_samples=50, centers=2, random_state=0, cluster_std=0.60)
# fit the model
clf = SGDClassifier(loss='hinge', alpha=0.01, max_iter=200,
                    fit_intercept=True, tol=1e-3)
clf.fit(X, Y)
# plot the line, the points and the nearest vectors to the plane
xx, yy = np.meshgrid(np.linspace(-1, 1, 10), np.linspace(-1, 1, 10))
Z = np.empty(X[0].shape)
for i, j in np.ndenumerate(X[0]):
    x1 = xx[i, j]
    x2 = xx[i, j]
    p = clf.decision_function(x1, x2)
    Z[i, j] = p[0]
levels = [0, 0.5, 1]
linestyles = ['dashed', 'solid', 'dashed']
colors = ['k', 'r', 'b']
plt.contour(X[0], X[1], Z, levels=levels, colors=colors, linestyles=linestyles)
plt.scatter(X[0], X[1], c=Y, cmap=plt.cm.Paired)
plt.axis('tight')
plt.show()
Total running time of the script: 0 minutes 00.20 seconds
Note: Click here to download the full example code
5184 SGD convex loss functions
A plot that compares the various convex loss functions supported by sklearn.linear_model
SGDClassifier
518 Generalized Linear Models 1135
```

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
defmodifiedhuberlossytrue ypred
z ypred ytrue
loss 4 z
lossz 1 1 zz 1 2
lossz 1 0
returnloss
xmin xmax 4 4
xx nplinspacexmin xmax 100
lw 2
pltplotxmin 0 0 xmax 1 1 0 0 colorgold lwlw
labelZeroone loss
pltplotxx npwherexx 1 1 xx 0 colorteal lwlw
labelHinge loss
pltplotxx npminimumxx 0 coloryellowgreen lwlw
labelPerceptron loss
pltplotxx nplog21 npexpxx colorcornflowerblue lwlw
labelLog loss
pltplotxx npwherexx 1 1 xx 0 2 colororange lwlw
1136 Chapter 5 Examples
```

scikitlearn user guide Release 0213

```
labelSquared hinge loss
pltplotxx modifiedhuberlossxx 1 colordarkorchid lwlw
linestyle labelModified Huber loss
pltylim0 8
pltlegendlocupper right
pltxlabelrDecision function fx
pltylabelLy1 fx
pltshow
```

Total running time of the script 0 minutes 0019 seconds

Note Click here to download the full example code

5185 Ordinary Least Squares and Ridge Regression Variance

Due to the few points in each dimension and the straight line that linear regression uses to follow these points as well as it can noise on the observations will cause great variance as shown in the first plot Every line’s slope can vary quite a bit for each prediction due to the noise induced in the observations
Ridge regression is basically minimizing a penalised version of the leastsquared function The penalising shrinks the value of the regression coefficients Despite the few data points in each dimension the slope of the prediction is much more stable and the variance in the line itself is greatly reduced in comparison to that of the standard linear regression

-
- 518 Generalized Linear Models 1137

```
•
printdoc
Code source Gaël Varoquaux
Modified for documentation by Jaques Grobler
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linearmodel
Xtrain = np.c5 1T
ytrain = 5 1
Xtest = np.c0 2T
np.random.seed(0)
classifiers = dict(zip(linearmodel.LinearRegression,
ridgelinearmodel.Ridge(alpha=1),
forname = clf in classifiers.items):
fig, ax = plt.subplots(figsize=(4, 3))
for i in range(6):
thisX = 1 * np.random.randn(size(2, 1) * Xtrain
clf.fit(thisX, ytrain)
ax.plot(Xtest, clf.predict(Xtest), color='gray')
ax.scatter(thisX, ytrain, s=3, color='gray', marker='o', zorder=10)
clf.fit(Xtrain, ytrain)
ax.plot(Xtest, clf.predict(Xtest), linewidth=2, color='blue')
ax.scatter(Xtrain, ytrain, s=30, color='red', marker='o', zorder=10)
ax.set_title(name)
ax.set_xlim(0, 2)
1138 Chapter 5 Examples
```


scikitlearn user guide Release 0213

axsetylim0 16

axsetxlabelX

axsetylabely

figtightlayout

pltshow

Total running time of the script 0 minutes 0130 seconds

Note Click here to download the full example code

5186 Plot Ridge coefficients as a function of the L2 regularization

Ridge Regression is the estimator used in this example Each color in the left plot represents one different dimension of the coefficient vector and this is displayed as a function of the regularization parameter The right plot shows how exact the solution is This example illustrates how a well defined solution is found by Ridge regression and how regularization affects the coefficients and their values The plot on the right shows how the difference of the coefficients from the estimator changes as a function of regularization

In this example the dependent variable Y is set as a function of the input features $y = Xw + c$ The coefficient vector w is randomly sampled from a normal distribution whereas the bias term c is set to a constant

As α tends toward zero the coefficients found by Ridge regression stabilize towards the randomly sampled vector w For big α strong regularisation the coefficients are smaller eventually converging at 0 leading to a simpler and biased solution These dependencies can be observed on the left plot

The right plot shows the mean squared error between the coefficients found by the model and the chosen vector w Less regularised models retrieve the exact coefficients error is equal to 0 stronger regularised models increase the error

Please note that in this example the data is nonnoisy hence it is possible to extract the exact coefficients

Author Kornel Kielczewski kornelkplusnetpl

printdoc

import matplotlib.pyplot as plt

import numpy as np

from sklearn.datasets import makeregression

from sklearn.linear_model import Ridge

518 Generalized Linear Models 1139

```
scikitlearn user guide Release 0213
from sklearn.metrics import mean_squared_error
clf = Ridge
X, y, w = make_regression(n_samples=10, n_features=10, coef=True,
                           random_state=1, bias=35)
coefs = []
errors = []
alphas = np.logspace(-6, 6, 200)
# Train the model with different regularisation strengths
for alpha in alphas:
    clf.set_params(alpha=alpha)
    clf.fit(X, y)
    coefs.append(clf.coef_)
    errors.append(mean_squared_error(clf.coef_, w))
# Display results
plt.figure(figsize=(20, 6))
plt.subplot(1, 2, 1)
ax = plt.gca()
ax.plot(alphas, coefs)
ax.set_xscale('log')
plt.xlabel('alpha')
plt.ylabel('weights')
plt.title('Ridge coefficients as a function of the regularization')
plt.axis('tight')
plt.subplot(1, 2, 2)
ax = plt.gca()
ax.plot(alphas, errors)
ax.set_xscale('log')
plt.xlabel('alpha')
plt.ylabel('error')
plt.title('Coefficient error as a function of the regularization')
plt.axis('tight')
plt.show()
Total running time of the script: 0 minutes 0.126 seconds
Note: Click here to download the full example code
5187 SGD Penalties
Contours of where the penalty is equal to 1 for the three penalties L1, L2 and elasticnet
All of the above are supported by sklearn.linear_model.StochasticGradientDescent
1140 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
l1color = navy
l2color = c
elasticnetcolor = darkorange
line = nplinspace(15, 15, 1001)
xx, yy = np.meshgrid(line, line)
l2 = xx**2 - yy**2
l1 = np.abs(xx) - np.abs(yy)
rho = 0.5
518 Generalized Linear Models 1141
```

```
scikitlearn user guide Release 0213
elasticnet rho l1 1 rho l2
pltfigurefigsize10 10 dpi100
ax pltgca
elasticnetcontour pltcontourxx yy elasticnet levels1
colorselasticnetcolor
l2contour pltcontourxx yy l2 levels1 colorsl2color
l1contour pltcontourxx yy l1 levels1 colorsl1color
axsetaspectequal
axspinesleftsetpositioncenter
axspinesrightsetcolornone
axspinesbottomsetpositioncenter
axspinesstopsetcolornone
pltclabelelasticnetcontour inline1 fontsize18
fmt10 elasticnet manual1 1
pltclabell2contour inline1 fontsize18
fmt10 L2 manual1 1
pltclabell1contour inline1 fontsize18
fmt10 L1 manual1 1
plttightlayout
pltshow
Total running time of the script 0 minutes 0247 seconds
Note Click here to download the full example code
5188 Regularization path of L1 Logistic Regression
Train l1penalized logistic regression models on a binary classification problem derived from the Iris dataset
The models are ordered from strongest regularized to least regularized The 4 coefficients of the models are collected
and plotted as a “regularization path” on the lefthand side of the figure strong regularizers all the coefficients are
exactly 0 When regularization gets progressively looser coefficients can get nonzero values one after the other
Here we choose the SAGA solver because it can efficiently optimize for the Logistic Regression loss with a non
smooth sparsity inducing l1 penalty
Also note that we set a low value for the tolerance to make sure that the model has converged before collecting the
coefficients
We also use warmstartTrue which means that the coefficients of the models are reused to initialize the next model
fit to speedup the computation of the fullpath
1142 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
Out
Computing regularization path
This took 2008s
printdoc
  Author Alexandre Gramfort alexandregramfortinriafr
  License BSD (Binary Search Decision) 3 clause
from time import time
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn import datasets
from sklearn.svm import l1_min
iris = datasets.load_iris()
X = iris.data
y = iris.target
518 Generalized Linear Models 1143
```

scikitlearn user guide Release 0213

```
X Xy 2
y yy 2
X Xmax Normalize X to speedup convergence
```

```
Demo path functions
cs l1mincX y losslog nplogspace0 7 16
printComputing regularization path
start time
clf linearmodelLogisticRegressionpenaltyl1 solversaga
tol1e6 maxiterint1e6
warmstartTrue
coefs
forcincs
clfsetparamsCc
clffitX y
coefsappendclfcfepravelcopy
printThis took 03fs time start
coefs nparraycoefs
pltplotnplog10cs coefs markero
ymin ymax pltylim
pltxlabellogC
pltylabelCoefficients
plttitleLogistic Regression Path
pltaxistight
pltshow
```

Total running time of the script 0 minutes 2022 seconds

Note Click here to download the full example code

5189 Polynomial interpolation

This example demonstrates how to approximate a function with a polynomial of degree ndegree by using ridge regression Concretely from nsamples 1d points it suffices to build the Vandermonde matrix which is nsamples x ndegree1 and has the following form

$$1 \ x_1 \ x_1^2 \ x_1^3 \ \dots \ 1 \ x_2 \ x_2^2 \ x_2^3 \ \dots$$

Intuitively this matrix can be interpreted as a matrix of pseudo features the points raised to some power The matrix is akin to but different from the matrix induced by a polynomial kernel

This example shows that you can do nonlinear regression with a linear model using a pipeline to add nonlinear features Kernel methods extend this idea and can induce very high even infinite dimensional feature spaces

```
scikitlearn user guide Release 0213
printdoc
  Author Mathieu Blondel
  Jake Vanderplas
  License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import Ridge
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
def fx
  function to approximate by polynomial interpolation
return x np sin x
generate points used to plot
xplot np.linspace(0, 10, 100)
generate points and keep a subset of them
x np.linspace(0, 10, 100)
rng np.random.RandomState(0)
rng.shuffle(x)
518 Generalized Linear Models 1145
```

```
scikitlearn user guide Release 0213
x np.sort(x, 2)
y fx
    create matrix versions of these arrays
X x np.newaxis
Xplot xplot np.newaxis
colors teal yellowgreen gold
lw 2
plt.plot(xplot, fxplot, color='cornflowerblue', linewidth=lw)
labelground truth
plt.scatter(x, y, color='navy', s=30, marker='o', label='training points')
for count, degree in enumerate(3, 4, 5):
    model = make_pipeline(PolynomialFeatures(degree), Ridge())
    model.fit(X, y)
    yplot = model.predict(Xplot)
    plt.plot(xplot, yplot, color='red', linewidth=lw)
labeldegree, d = degree
plt.legend(loc='lower left')
plt.show()
Total running time of the script: 0 minutes 00.20 seconds
Note: Click here to download the full example code
51810 Logistic function
Shown in the plot is how the logistic regression would in this synthetic dataset classify values as either 0 or 1 ie
class one or two using the logistic curve
1146 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
printdoc
Code source Gael Varoquaux
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linearmodel
from scipy.special import expit
General a toy datasets its just a straight line with some Gaussian noise
xmin xmax 5 5
nsamples 100
nprandomseed0
X nprandomnormalsizensamples
y X 0astypenpfloat
XX 0 4
X 3 nprandomnormalsizensamples
X X npnewaxis
Fit the classifier
clf linearmodelLogisticRegressionC1e5 solverlbfgs
clffitX y
and plot the result
pltfigure1 figsize4 3
pltclf
pltscatterXravel y colorblack zorder20
Xtest nplinspace5 10 300
loss expitXtest clfcoef clfintercepttravel
pltplotXtest loss colorred linewidth3
ols linearmodelLinearRegression
olsfitX y
pltplotXtest olscoef Xtest olsintercept linewidth1
pltaxhline5 color5
pltlabely
pltlabelX
pltxticksrange5 10
pltxticks0 05 1
pltylim25 125
pltxlim4 10
pltlegendLogistic Regression Model Linear Regression Model
loclower right fontssizesmall
plttightlayout
pltshow
Total running time of the script 0 minutes 0046 seconds
Note Click here to download the full example code
518 Generalized Linear Models 1147
```

```
scikitlearn user guide Release 0213
51811 SGD Weighted samples
Plot decision function of a weighted dataset where the size of points is proportional to its weight
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linearmodel
    we create 20 points
nprandomseed0
X nprnprandomrandn10 2 1 1 nprandomrandn10 2
y 1 10 1 10
sampleweight 100 npabsnprandomrandn20
    and assign a bigger weight to the last 10 samples
sampleweight10 10
    plot the weighted data points
xx yy npmeshgridnplinspace4 5 500 nplinspace4 5 500
pltfigure
pltscatterX 0 X 1 cy ssampleweight alpha09
cmappltcmbone edgecolorblack
    fit the unweighted model
clf linearmodelSGDClassifieralpha001 maxiter100 tol1e3
1148 Chapter 5 Examples
```

scikitlearn user guide Release 0213

clffitX y

Z clfdecisionfunctionnpcxxravel yyravel

Z Zreshapexxshape

noweights pltcontourxx yy Z levels0 linestylessolid

fit the weighted model

clf linearmodelSGDClassifieralpha001 maxiter100 tol1e3

clffitX y sampleweightssampleweight

Z clfdecisionfunctionnpcxxravel yyravel

Z Zreshapexxshape

samplesweights pltcontourxx yy Z levels0 linestyleddashed

pltlegendnoweightscollections0 samplesweightscollections0

no weights with weights loclower left

pltxticks

pltxticks

pltshow

Total running time of the script 0 minutes 0086 seconds

Note Click here to download the full example code

51812 Logistic Regression 3class Classifier

Show below is a logisticregression classifiers decision boundaries on the first two dimensions sepal length and width of the iris dataset The datapoints are colored according to their labels

printdoc

Code source Gaël Varoquaux

Modified for documentation by Jaques Grobler

License BSD (Binary Search Decision) 3 clause

518 Generalized Linear Models 1149

scikitlearn user guide Release 0213

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data[:, 2:] # we only take the first two features
Y = iris.target

logreg = LogisticRegression(C=1e5, solver='lbfgs', multi_class='multinomial')
# Create an instance of Logistic Regression Classifier and fit the data
logreg.fit(X, Y)

# Plot the decision boundary. For that we will assign a color to each
# point in the mesh [xmin, xmax, ymin, ymax]
xmin, xmax, ymin, ymax = X.min(), X.max(), Y.min(), Y.max()
h = 0.02 # step size in the mesh
xx, yy = np.meshgrid(np.arange(xmin, xmax, h), np.arange(ymin, ymax, h))
Z = logreg.predict(np.c_[xx.ravel(), yy.ravel()])
# Put the result into a color plot
Z = Z.reshape(xx.shape)
plt.figure(1, figsize=(4, 3))
plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)
# Plot also the training points
plt.scatter(X[:, 0], X[:, 1], c=Y, edgecolors='k', cmap=plt.cm.Paired)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.xlim(xmin, xmax)
plt.ylim(ymin, ymax)
plt.xticks(
plt.yticks(
plt.show
```

Total running time of the script: 0 minutes 0083 seconds

Note: [Click here to download the full example code](#)

51813 Linear Regression Example

This example uses the only the first feature of the diabetes dataset in order to illustrate a two-dimensional plot of this regression technique. The straight line can be seen in the plot showing how linear regression attempts to draw a straight line that will best minimize the residual sum of squares between the observed responses in the dataset and the responses predicted by the linear approximation. The coefficients, the residual sum of squares and the variance score are also calculated.

scikitlearn user guide Release 0213
Out
Coefficients
93823786125
Mean squared error 254807
Variance score 047
printdoc
Code source Jaques Grobler
License BSD (Binary Search Decision) 3 clause
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets linearmodel
from sklearnmetrics import meansquarederror r2score
Load the diabetes dataset
diabetes datasetsloaddiabetes
518 Generalized Linear Models 1151

```
scikitlearn user guide Release 0213
Use only one feature
diabetesX diabetesdata npnewaxis 2
Split the data into trainingtesting sets
diabetesXtrain diabetesX20
diabetesXtest diabetesX20
Split the targets into trainingtesting sets
diabetesytrain diabetestarget20
diabetesytest diabetestarget20
Create linear regression object
regr linearmodelLinearRegression
Train the model using the training sets
regrfitdiabetesXtrain diabetesytrain
Make predictions using the testing set
diabetesypred regrpredictdiabetesXtest
The coefficients
printCoefficients n regrcoef
The mean squared error
printMean squared error 2f
meansquarederrordiabetesytest diabetesypred
Explained variance score 1 is perfect prediction
printVariance score 2f r2scorediabetesytest diabetesypred
Plot outputs
pltscatterdiabetesXtest diabetesytest colorblack
pltplotdiabetesXtest diabetesypred colorblue linewidth3
pltxticks
pltyticks
pltshow
Total running time of the script 0 minutes 0175 seconds
Note Click here to download the full example code
51814 Robust linear model estimation using RANSAC
In this example we see how to robustly fit a linear model to faulty data using the RANSAC algorithm
1152 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
Out
Estimated coefficients true linear regression RANSAC
821903908407869 5417236387 8208533159
import numpy as np
from matplotlib import pyplot as plt
from sklearn import linear_model datasets
nsamples 1000
noutliers 50
X y coef datasets make_regression nsamples nsamples nfeatures 1
ninformative 1 noise 10
coef True random_state 0
Add outlier data
np.random.seed(0)
518 Generalized Linear Models 1153
```

```
scikitlearn user guide Release 0213
Xnoutliers 3 05 nprandomnormalsizenoutliers 1
ynoutliers 3 10 nprandomnormalsizenoutliers
Fit line using all data
lr linearmodelLinearRegression
lrfitX y
Robustly fit linear model with RANSAC algorithm
ransac linearmodelRANSACRegressor
ransacfitX y
inliermask ransacinliermask
outliermask nplogicalnotinliermask
Predict data of estimated models
lineX nparangeXmin Xmax npnewaxis
liney lrpredictlineX
lineyransac ransacpredictlineX
Compare estimated coefficients
printEstimated coefficients true linear regression RANSAC
printcoef lrcoef ransacestimatorcoef
lw 2
pltscatterXinliermask yinliermask coloryellowgreen marker
labelInliers
pltscatterXoutliermask youtliermask colorgold marker
labelOutliers
pltplotlineX liney colornavy linewidthlw labelLinear regressor
pltplotlineX lineyransac colorcornflowerblue linewidthlw
labelRANSAC regressor
pltlegendlocclower right
pltxlabelInput
pltylabelResponse
pltshow
Total running time of the script 0 minutes 0028 seconds
Note Click here to download the full example code
51815 Sparsity Example Fitting only features 1 and 2
Features 1 and 2 of the diabetesdataset are fitted and plotted below It illustrates that although feature 2 has a strong
coefficient on the full model it does not give us much regarding ywhen compared to just feature 1
1154 Chapter 5 Examples
```


scikitlearn user guide Release 0213

-
-

518 Generalized Linear Models 1155

scikitlearn user guide Release 0213

```
•
printdoc
Code source Gaël Varoquaux
Modified for documentation by Jaques Grobler
License BSD (Binary Search Decision) 3 clause
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
from sklearn import datasets, linear_model
diabetes = datasets.load_diabetes()
indices = 0:1
X_train = diabetes.data[indices]
X_test = diabetes.data[indices]
y_train = diabetes.target[indices]
y_test = diabetes.target[indices]
ols = linear_model.LinearRegression()
ols.fit(X_train, y_train)
```

```
Plot the figure
def plot_fig(fignum, elev, azim, X_train, clf):
    fig = plt.figure(figsize=(4, 3))
    plt.clf()
    ax = Axes3D(fig, elev, azim)
    ax.scatter(X_train[:, 0], X_train[:, 1], y_train, c='k', marker='o')
    ax.plot_surface(np.linspace(1, 15, 15), np.linspace(1, 15, 15),
                    clf.predict(np.linspace(1, 15, 15), np.linspace(1, 15, 15)).T),
                    zorder=2)
    plt.show()
1156 Chapter 5 Examples
```

scikitlearn user guide Release 0213

alpha5
axsetxlabelX1
axsetylabelX2
axsetxlabelY
axwxaxissetticklabels
axwyaxissetticklabels
axwzaxissetticklabels

Generate the three different figures from different views

elev 435
azim 110
plotfigs1 elev azim Xtrain ols
elev 5
azim 0
plotfigs2 elev azim Xtrain ols
elev 5
azim 90
plotfigs3 elev azim Xtrain ols
pltshow

Total running time of the script 0 minutes 0197 seconds

Note [Click here to download the full example code](#)

51816 Lasso on dense and sparse data

We show that linearmodelLasso provides the same results for dense and sparse data and that in the case of sparse data the speed is improved

Out
Dense matrices
Sparse Lasso done in 0186067s
Dense Lasso done in 0034536s
Distance between coefficients 9043732562018544e14
Sparse matrices
Matrix density 062630000000000001
Sparse Lasso done in 0284597s
Dense Lasso done in 0955330s
Distance between coefficients 7344760355532163e12

printdoc
from time import time
from scipy import sparse
from scipy import linalg
518 Generalized Linear Models 1157

```
scikitlearn user guide Release 0213
from sklearn.datasets.samples_generator import make_regression
from sklearn.linear_model import Lasso
```

```

The two Lasso implementations on Dense data
print Dense matrices
X, y = make_regression(n_samples=200, n_features=5000, random_state=0)
X_sparse = sparse.coo_matrix(X)
alpha = 1
sparse_lasso = Lasso(alpha=alpha, fit_intercept=False, max_iter=1000)
dense_lasso = Lasso(alpha=alpha, fit_intercept=False, max_iter=1000)
t0 = time
sparse_lasso.fit(X_sparse, y)
print Sparse Lasso done in %fs %time %t0
t0 = time
dense_lasso.fit(X, y)
print Dense Lasso done in %fs %time %t0
print Distance between coefficients: %s
      l1 norm sparse_lasso.coef_ dense_lasso.coef_
```

```

The two Lasso implementations on Sparse data
print Sparse matrices
Xs = X.copy()
Xs[Xs == 0] = 0.0
Xs_sparse = sparse.coo_matrix(Xs)
Xs_Xstocsc = Xs_Xstocsc
print Matrix density: %s %Xs.nnz / float(Xs.size) 100
alpha = 0.1
sparse_lasso = Lasso(alpha=alpha, fit_intercept=False, max_iter=10000)
dense_lasso = Lasso(alpha=alpha, fit_intercept=False, max_iter=10000)
t0 = time
sparse_lasso.fit(Xs_sparse, y)
print Sparse Lasso done in %fs %time %t0
t0 = time
dense_lasso.fit(Xstocsc, y)
print Dense Lasso done in %fs %time %t0
print Distance between coefficients: %s
      l1 norm sparse_lasso.coef_ dense_lasso.coef_
Total running time of the script: 0 minutes 15.65 seconds
Note: Click here to download the full example code
1158 Chapter 5 Examples
```

scikitlearn user guide Release 0213

51817 HuberRegressor vs Ridge on dataset with strong outliers

Fit Ridge and HuberRegressor on a dataset with outliers

The example shows that the predictions in ridge are strongly influenced by the outliers present in the dataset The Huber regressor is less influenced by the outliers since the model uses the linear loss for these As the parameter epsilon is increased for the Huber regressor the decision function approaches that of the ridge

Authors Manoj Kumar mks542nyuedu

License BSD (Binary Search Decision) 3 clause

printdoc

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import makeregression

from sklearn.linear_model import HuberRegressor Ridge

Generate toy data

rng = np.random.RandomState(0)

X, y = makeregression(n_samples=20, n_features=1, random_state=0, noise=40,

bias=1000)

Add four strong outliers to the dataset

X_outliers = rng.normal(0, 0.5, size=4, 1)

518 Generalized Linear Models 1159

scikitlearn user guide Release 0213

youtliers rngnormal0 20 size4

Xoutliers2 Xmax Xmean 4

Xoutliers2 Xmin Xmean 4

youtliers2 ymin ymean 4

youtliers2 ymax ymean 4

X npvstackX Xoutliers

y npconcatenatey youtliers

pltplotX y b

Fit the huber regressor over a series of epsilon values

colors r b y m

x np.linspaceXmin Xmax 7

epsilonvalues 135 15 175 19

fork epsilon in enumerateepsilonvalues

huber HuberRegressorfitinterceptTrue alpha00 maxiter100

epsilonepsilon

huberfitX y

coef hubercoef x huberintercept

pltplotx coef colorsk labelhuber loss s epsilon

Fit a ridge regressor to compare it to huber regressor

ridge RidgefitinterceptTrue alpha00 randomstate0 normalizeTrue

ridgefitX y

coefridge ridgecoef

coef ridgecoef x ridgeintercept

pltplotx coef g labelridge regression

plttitleComparison of HuberRegressor vs Ridge

pltxlabelX

pltylabely

pltlegendloc0

pltshow

Total running time of the script 0 minutes 0032 seconds

Note Click here to download the full example code

51818 Joint feature selection with multitask Lasso

The multitask lasso allows to fit multiple regression problems jointly enforcing the selected features to be the same

across tasks This example simulates sequential measurements each task is a time instant and the relevant features

vary in amplitude over time while being the same The multitask lasso imposes that features that are selected at one

time point are select for all time point This makes feature selection by the Lasso more stable

1160 Chapter 5 Examples

scikitlearn user guide Release 0213

-
-

518 Generalized Linear Models 1161

```
scikitlearn user guide Release 0213
printdoc
  Author Alexandre Gramfort alexandregramfortinriafr
  License BSD (Binary Search Decision) 3 clause
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import MultiTaskLasso Lasso
rng = np.random.RandomState(42)
  Generate some 2D coefficients with sine waves with random frequency and phase
nsamples, nfeatures, ntasks = 100, 30, 40
nrelevantfeatures = 5
coef = np.zeros((ntasks, nfeatures))
times = np.linspace(0, 2, np.pi * ntasks)
for k in range(nrelevantfeatures):
    coef[k, :] = np.sin(1 + rng.randn(1) * times) * 3 * rng.randn(1)
X = rng.randn(nsamples, nfeatures)
Y = np.dot(X, coef.T)
rng.randn(nsamples, ntasks)
coef_lasso = np.array(Lasso(alpha=0.5).fit(X, Y).coef_)
coef_multitask_lasso = MultiTaskLasso(alpha=1).fit(X, Y).coef_
```

```
Plot support and time series
fig = plt.figure(figsize=(8, 5))
plt.subplot(1, 2, 1)
plt.spy(coef_lasso)
plt.xlabel('Feature')
plt.ylabel('Time or Task')
plt.text(10, 5, 'Lasso')
plt.subplot(1, 2, 2)
plt.spy(coef_multitask_lasso)
plt.xlabel('Feature')
plt.ylabel('Time or Task')
plt.text(10, 5, 'MultiTaskLasso')
fig.suptitle('Coefficient nonzero locations')
feature_toplot = 0
plt.figure()
lw = 2
plt.plot(coef, feature_toplot, color='seagreen', linewidth=lw)
label = 'Ground truth'
plt.plot(coef_lasso, feature_toplot, color='cornflowerblue', linewidth=lw)
label = 'Lasso'
plt.plot(coef_multitask_lasso, feature_toplot, color='gold', linewidth=lw)
label = 'MultiTaskLasso'
plt.legend(loc='upper center')
plt.axis('tight')
plt.ylim(1, 11)
plt.show()
Total running time of the script: 0 minutes 00.61 seconds
1162 Chapter 5: Examples
```


scikitlearn user guide Release 0213
Note Click here to download the full example code
51819 Comparing various online solvers
An example showing how different online solvers perform on the handwritten digits dataset
Out
training SGD
training ASGD
training Perceptron
training PassiveAggressive I
training PassiveAggressive II
training SAG
Author Rob Zinkov rob at zinkov dot com
License BSD (Binary Search Decision) 3 clause
518 Generalized Linear Models 1163

```
scikitlearn user guide Release 0213
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.linear_model import SGDClassifier, Perceptron
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.linear_model import LogisticRegression

heldout = 0.95
n_train = 0.05
n_test = 0.05
n_val = 0.05
n_rounds = 20
digits = datasets.load_digits()
X, y = digits.data, digits.target
classifiers = [
    SGDClassifier(max_iter=100, tol=1e-3),
    SGDClassifier(average=True, max_iter=1000, tol=1e-3),
    Perceptron(tol=1e-3),
    PassiveAggressiveClassifier(loss='hinge', C=10, tol=1e-4),
    PassiveAggressiveClassifier(loss='squared_hinge', C=10, tol=1e-4),
    LogisticRegression(solver='sag', tol=1e-1, C=1e4, X_shape=0,
                        multi_class='auto')
]

xx = 1
n_train, n_test, n_val = 0.95, 0.05, 0.05
for name, clf in zip(classifiers, classifiers):
    print('Training %s' % name)
    rng = np.random.RandomState(42)
    yy = rng.randn(n_train)
    for i in range(n_train):
        yy[i] = 0
    for i in range(n_test):
        yy[i] = 1
    for i in range(n_val):
        yy[i] = 2
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=rng)
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    yy.append(1 - np.mean(y_pred == y_test))
    yy.append(np.mean(y))
plt.plot(xx, yy, label='name')
plt.legend(loc='upper right')
plt.xlabel('Proportion train')
plt.ylabel('Test Error Rate')
plt.show()

Total running time of the script: 0 minutes 24.266 seconds
Note: Click here to download the full example code
51820 Orthogonal Matching Pursuit
Using orthogonal matching pursuit for recovering a sparse signal from a noisy measurement encoded with a dictionary
1164 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdoc
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import OrthogonalMatchingPursuit
from sklearn.linear_model import OrthogonalMatchingPursuitCV
from sklearn.datasets import make_sparse_coded_signal
ncomponents, nfeatures = 512, 100
nnonzero_coefs = 17
generate the data
y, Xw
x0, nnonzero_coefs
518 Generalized Linear Models 1165
```

```
scikitlearn user guide Release 0213
y X w makesparsecodedsignalnsamples1
ncomponentsncomponents
nfeaturesnfeatures
nnonzerocoefsnnonzerocoefs
randomstate0
idx wnonzero
    distort the clean signal
ynoisyy 005 nprandomrandnleny
plot the sparse signal
pltfigurefigsize7 7
pltsubplot4 1 1
pltxlim0 512
plttitleSparse signal
pltstemidx widx
    plot the noise-free reconstruction
omp OrthogonalMatchingPursuitnnonzerocoefsnnonzerocoefs
ompfitX y
coef ompcoef
idxr coefnonzero
pltsubplot4 1 2
pltxlim0 512
plttitleRecovered signal from noise-free measurements
pltstemidxr coefidxr
    plot the noisy reconstruction
ompfitX ynoisy
coef ompcoef
idxr coefnonzero
pltsubplot4 1 3
pltxlim0 512
plttitleRecovered signal from noisy measurements
pltstemidxr coefidxr
    plot the noisy reconstruction with number of nonzeros set by CV (Cross-Validation)
ompcv OrthogonalMatchingPursuitCVcv5
ompcvfitX ynoisy
coef ompcvcoef
idxr coefnonzero
pltsubplot4 1 4
pltxlim0 512
plttitleRecovered signal from noisy measurements with CV (Cross-Validation)
pltstemidxr coefidxr
pltsubplotsadjust006 004 094 090 020 038
pltstitleSparse signal recovery with Orthogonal Matching Pursuit
fontsize16
pltshow
Total running time of the script 0 minutes 0499 seconds
Note Click here to download the full example code
1166 Chapter 5 Examples
```

scikitlearn user guide Release 0213

51821 MNIST classification using multinomial logistic L1

Here we fit a multinomial logistic regression with L1 penalty on a subset of the MNIST digits classification task We use the SAGA algorithm for this purpose this a solver that is fast when the number of samples is significantly larger than the number of features and is able to finely optimize nonsmooth objective functions which is the case with the l1penalty Test accuracy reaches 08 while weight vectors remains sparse and therefore more easily interpretable Note that this accuracy of this l1penalized linear model is significantly below what can be reached by an l2penalized linear model or a nonlinear multilayer perceptron model on this dataset

Out

Sparsity with L1 penalty 8080

Test score with L1 penalty 08351

Example run in 28654 s

```
import time
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import fetchopenml
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.utils import check_random_state
printdoc
    Author Arthur Mensch arthurmenschm4xorg
    License BSD (Binary Search Decision) 3 clause
518 Generalized Linear Models 1167
```

```
scikitlearn user guide Release 0213
Turn down for faster convergence
t0 = time.time()
train_samples = 5000
Load data from https://www.openml.org/d/554
X, y = fetch_openml(mnist784, version=1, return_X_y=True)
random_state = check_random_state(0)
permutation = random_state.permutation(X.shape[0])
X = X[permutation]
y = y[permutation]
X = X.reshape(X.shape[0], 1)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, train_size=train_samples, test_size=10000)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
Turn up tolerance for faster convergence
clf = LogisticRegression(C=50, train_samples=
    multiclass_multinomial,
    penalty='l1', solver='saga', tol=0.1)
clf.fit(X_train, y_train)
sparsity = np.mean(clf.coef_ == 0)
score = clf.score(X_test, y_test)
print('Best C: %f' % clf.C)
print('Sparsity with L1 penalty: %f' % sparsity)
print('Test score with L1 penalty: %f' % score)
coef = clf.coef_.copy()
plt.figure(figsize=(10, 5))
scale = np.abs(coef).max()
for i in range(10):
    l1plot = plt.subplot(2, 5, i + 1)
    l1plot.imshow(coef[i].reshape(28, 28), interpolation='nearest',
        cmap=plt.cm.RdBu, vmin=-scale, vmax=scale)
    l1plot.set_xticks()
    l1plot.set_yticks()
    l1plot.set_xlabel('Class %i' % i)
    plt.suptitle('Classification vector for %i' % i)
runtime = time.time() - t0
print('Example run in %fs' % runtime)
plt.show()
Total running time of the script: 0 minutes 28.655 seconds
Note: Click here to download the full example code
51822 Plot multiclass SGD on the iris dataset
Plot decision surface of multiclass SGD on iris dataset
The hyperplanes corresponding to the three one-versus-all
OV-A classifiers are represented by the dashed lines
1168 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.linear_model import SGDClassifier
import some data to play with
iris = datasets.load_iris()
we only take the first two features We could
avoid this ugly slicing by using a twodim dataset
X = iris.data[:, 2:]
y = iris.target
colors = ['r', 'b', 'g']
shuffle
idx = np.arange(X.shape[0])
np.random.seed(13)
np.random.shuffle(idx)
X = X[idx]
y = y[idx]
standardize
mean = X.mean(axis=0)
std = X.std(axis=0)
518 Generalized Linear Models 1169
```

scikitlearn user guide Release 0213

X X mean std

h 02 step size in the mesh

clf SGDClassifier(alpha=0.001, max_iter=100, tol=1e-3, fit_X, y

create a mesh to plot in

xmin xmax X 0min 1 X 0max 1

ymin ymax X 1min 1 X 1max 1

xx yy np.meshgrid(nparange(xmin, xmax, h

nparange(ymin, ymax, h

Plot the decision boundary For that we will assign a color to each

point in the mesh xmin xmax, ymin ymax

Z clf.predict(xx.ravel(), yy.ravel())

Put the result into a color plot

Z Z.reshape(xx.shape)

cs plt.contourf(xx, yy, Z, cmap=plt.cm.Paired

plt.axis('tight')

Plot also the training points

for i, color in zip(clf.classes_, colors)

idx, np.where(y == i)

plt.scatter(X[idx, 0], X[idx, 1], c=color, label=i, s=20)

cmap=plt.cm.Paired, edgecolor='black', s=20)

plt.title('Decision surface of multiclass SGD

plt.axis('tight')

Plot the three one-against-all classifiers

xmin, xmax, plt.xlim

ymin, ymax, plt.ylim

coef = clf.coef

intercept = clf.intercept

def plot_hyperplane(c, color):

def line(x0,

return x0 * coef[c, 0] + intercept[c] - coef[c, 1]

plt.plot(xmin, xmax, line(xmin, xmax),

ls='color', color=

for i, color in zip(clf.classes_, colors)

plot_hyperplane(i, color)

plt.legend

plt.show

Total running time of the script: 0 minutes 00.50 seconds

Note: Click here to download the full example code

1170 Chapter 5 Examples

scikitlearn user guide Release 0213

51823 Lasso and Elastic Net for Sparse Signals

Estimates Lasso and ElasticNet regression models on a manually generated sparse signal corrupted with an additive noise Estimated coefficients are compared with the groundtruth

Out

Lassoalpha01

r2 on test data 0658064

ElasticNetalpha01 l1ratio07

r2 on test data 0642515

printdoc

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import r2score

Generate some sparse data to play with

518 Generalized Linear Models 1171

```
scikitlearn user guide Release 0213
nprandomseed42
nsamples nfeatures 50 100
X nprandomrandnnsamples nfeatures
Decreasing coef w alternated signs for visualization
idx nparangenfeatures
coef 1 idxnpexpidx 10
coef10 0 sparsify coef
y npdotX coef
Add noise
y 001 nprandomnormalsizensamples
Split data in train set and test set
nsamples Xshape0
Xtrain ytrain Xnsamples 2 ynsamples 2
Xtest ytest Xnsamples 2 ynsamples 2

Lasso
from sklearnlinear import Lasso
alpha 01
lasso Lassoalphaalpha
ypredlasso lassofitXtrain ytrainpredictXtest
r2scorelasso r2scoreytest ypredlasso
printlasso
printr2 on test data f r2scorelasso

ElasticNet
from sklearnlinear import ElasticNet
enet ElasticNetalphaalpha l1ratio07
ypreden et enetfitXtrain ytrainpredictXtest
r2scoreenet r2scoreytest ypredenet
printenet
printr2 on test data f r2scoreenet
m s pltstemnpwhereenetcoef0 enetcoefenetcoef 0
markerfmtx labelElastic net coefficients
pltsetpm s color2ca02c
m s pltstemnpwherelassocoef0 lassocoeflassocoef 0
markerfmtx labelLasso coefficients
pltsetpm s colorff7f0e
pltstemnpwherecoef0 coefcoef 0 labeltrue coefficients
markerfmtbx
pltlegendlocbest
plttitleLasso R2 3f Elastic Net R2 3f
r2scorelasso r2scoreenet
pltshow
Total running time of the script 0 minutes 0071 seconds
1172 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Note Click here to download the full example code

51824 TheilSen Regression

Computes a TheilSen Regression on a synthetic dataset

SeeTheilSen estimator generalizedmedianbased estimator for more information on the regressor

Compared to the OLS ordinary least squares estimator the TheilSen estimator is robust against outliers It has a breakdown point of about 293 in case of a simple linear regression which means that it can tolerate arbitrary corrupted data outliers of up to 293 in the twodimensional case

The estimation of the model is done by calculating the slopes and intercepts of a subpopulation of all possible combinations of p subsample points If an intercept is fitted p must be greater than or equal to nfeatures + 1 The final slope and intercept is then defined as the spatial median of these slopes and intercepts

In certain cases TheilSen performs better than RANSAC which is also a robust method This is illustrated in the second example below where outliers with respect to the xaxis perturb RANSAC Tuning the residualthreshold

parameter of RANSAC remedies this but in general a priori knowledge about the data and the nature of the outliers

is needed Due to the computational complexity of TheilSen it is recommended to use it only for small problems in

terms of number of samples and features For larger problems the maxsubpopulation parameter restricts the

magnitude of all possible combinations of p subsample points to a randomly chosen subset and therefore also limits the

runtime Therefore TheilSen is applicable to larger problems with the drawback of losing some of its mathematical

properties since it then works on a random subset

•

518 Generalized Linear Models 1173

scikitlearn user guide Release 0213

```
•
  Author Florian Wilhelm florianwilhelmgmailcom
  License BSD (Binary Search Decision) 3 clause
import time
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression TheilSenRegressor
from sklearn.linear_model import RANSACRegressor
printdoc
estimators OLS LinearRegression
TheilSen TheilSenRegressorrandomstate42
RANSAC RANSACRegressorrandomstate42
colors OLS turquoise TheilSen gold RANSAC lightgreen
lw 2
```

```
Outliers only in the y direction
nprandomseed0
nsamples 200
Linear model  $y = 3x + N(0, 1)$ 
x nprandomrandnnsamples
w 3
c 2
noise 0.1 nprandomrandnnsamples
1174 Chapter 5 Examples
```

scikitlearn user guide Release 0213

```
y wx c noise
10 outliers
y20 20 x20
X x npnewaxis
pltscatterx y colorindigo markerx s40
linex nparray3 3
forname estimator inestimators
t0 timetime
estimatorfitX y
elapsedtime timetime t0
ypred estimatorpredictlinexreshape2 1
pltplotlinex ypred colorcolorsname linewidthlw
labelsfit time 2fs name elapsedtime
pltaxistight
pltlegendlocupper left
plttitleCorrupt y
```

```
Outliers in the X direction
nprandomseed0
Linear model y 3 x N2 01 2
x nprandomrandnnsamples
noise 01 nprandomrandnnsamples
y 3x 2 noise
10 outliers
x20 99
y20 22
X x npnewaxis
pltfigure
pltscatterx y colorindigo markerx s40
linex nparray3 10
forname estimator inestimators
t0 timetime
estimatorfitX y
elapsedtime timetime t0
ypred estimatorpredictlinexreshape2 1
pltplotlinex ypred colorcolorsname linewidthlw
labelsfit time 2fs name elapsedtime
pltaxistight
pltlegendlocupper left
plttitleCorrupt x
pltshow
Total running time of the script 0 minutes 1359 seconds
Note Click here to download the full example code
518 Generalized Linear Models 1175
```

scikitlearn user guide Release 0213

51825 Plot multinomial and OnevsRest Logistic Regression

Plot decision surface of multinomial and OnevsRest Logistic Regression The hyperplanes corresponding to the three OnevsRest OVR classifiers are represented by the dashed lines

-

1176 Chapter 5 Examples

scikitlearn user guide Release 0213

•

Out

training score 0995 multinomial

training score 0976 ovr

printdoc

Authors Tom Dupre la Tour tomduprelatourm4xorg

License BSD (Binary Search Decision) 3 clause

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import makeblobs

from sklearn.linear_model import LogisticRegression

make 3class dataset for classification

centers 5 0 0 15 5 1

X y makeblobsnsamples1000 centerscenters randomstate40

transformation 04 02 04 12

X npdotX transformation

formulticlass inmultinomial ovr

clf LogisticRegression solver sag maxiter100 randomstate42

518 Generalized Linear Models 1177

```
scikitlearn user guide Release 0213
multiclassmulticlassfitX y
print the training scores
printtraining score 3fs clfscoreX y multiclass
create a mesh to plot in
h 02 step size in the mesh
xmin xmax X 0min 1 X 0max 1
ymin ymax X 1min 1 X 1max 1
xx yy npmeshgridnparangexmin xmax h
nparangeymin ymax h
Plot the decision boundary For that we will assign a color to each
point in the mesh xmin xmaxxmin ymax
Z clfpredictnpcxxravel yy.ravel
Put the result into a color plot
Z Zshapexxshape
pltfigure
pltcontourfxx yy Z cmappltcmPaired
plttitleDecision surface of LogisticRegression s multiclass
pltaxistight
Plot also the training points
colors bry
fori color inzipclfclasses colors
idx npwherey i
pltscatterXidx 0 Xidx 1 ccolor cmappltcmPaired
edgecolorblack s20
Plot the three oneagainstall classifiers
xmin xmax pltxlim
ymin ymax pltylim
coef clfcoef
intercept clfintercept
defplothyperplanec color
deflinex0
returnx0coefc 0 interceptc coefc 1
pltplotxmin xmax linexmin linemax
ls colorcolor
fori color inzipclfclasses colors
plothyperplanei color
pltshow
Total running time of the script 0 minutes 0262 seconds
Note Click here to download the full example code
51826 Robust linear estimator fitting
Here a sine function is fit with a polynomial of order 3 for values close to zero
Robust fitting is demoed in different situations
1178 Chapter 5 Examples
```


scikitlearn user guide Release 0213

- No measurement errors only modelling errors fitting a sine with a polynomial
- Measurement errors in X
- Measurement errors in y

The median absolute deviation to non corrupt new data is used to judge the quality of the prediction

What we can see that

- RANSAC is good for strong outliers in the y direction
- TheilSen is good for small outliers both in direction X and y but has a break point above which it performs worse than OLS
- The scores of HuberRegressor may not be compared directly to both TheilSen and RANSAC because it does not attempt to completely filter the outliers but lessen their effect
-

518 Generalized Linear Models 1179

scikitlearn user guide Release 0213

-
-

1180 Chapter 5 Examples

scikitlearn user guide Release 0213

-
-

```
from matplotlib import pyplot as plt
import numpy as np
from sklearn.linear_model import
```

518 Generalized Linear Models 1181

```
scikitlearn user guide Release 0213
LinearRegression TheilSenRegressor RANSACRegressor HuberRegressor
from sklearnmetrics import meansquarederror
from sklearnpreprocessing import PolynomialFeatures
from sklearnpipeline import makepipeline
nprandomseed42
X nprandomnormalsize400
y npsinX
Make sure that it X is 2D
X X npnewaxis
Xtest nprandomnormalsize200
ytest npsinXtest
Xtest Xtest npnewaxis
yerrors ycopy
yerrors3 3
Xerrors Xcopy
Xerrors3 3
yerrorslarge ycopy
yerrorslarge3 10
Xerrorslarge Xcopy
Xerrorslarge3 10
estimators OLS LinearRegression
TheilSen TheilSenRegressorrandomstate42
RANSAC RANSACRegressorrandomstate42
HuberRegressor HuberRegressor
colors OLS turquoise TheilSen gold RANSAC lightgreen
↔HuberRegressor black
linestyle OLS TheilSen RANSAC HuberRegressor
lw 3
xplot nplinspaceXmin Xmax
fortitle thisX thisy in
Modeling Errors Only X y
Corrupt X Small Deviants Xerrors y
Corrupt y Small Deviants X yerrors
Corrupt X Large Deviants Xerrorslarge y
Corrupt y Large Deviants X yerrorslarge
pltfigurefigsize5 4
pltplotthisX 0 thisy b
forname estimator inestimators
model makepipelinePolynomialFeatures3 estimator
modelfitthisX thisy
mse meansquarederrormodelpredictXtest ytest
yplot modelpredictxplot npnewaxis
pltplotxplot yplot colorcolorsname linestylelinestylename
linewidthlw label s error 3f name mse
legendtitle Error of Mean nAbsolute Deviation nto Noncorrupt Data
legend pltlegendlocupper right frameonFalse titlelegendtitle
propdictsizeXsmall
1182 Chapter 5 Examples
```

scikitlearn user guide Release 0213
pltxlim4 102
pltylim2 102
plttitletitle
pltshow
Total running time of the script 0 minutes 3950 seconds
Note Click here to download the full example code
51827 L1 Penalty and Sparsity in Logistic Regression
Comparison of the sparsity percentage of zero coefficients of solutions when L1 L2 and ElasticNet penalty are used
for different values of C We can see that large values of C give more freedom to the model Conversely smaller values
of C constrain the model more In the L1 penalty case this leads to sparser solutions As expected the ElasticNet
penalty sparsity is between that of L1 and L2
We classify 8x8 images of digits into two classes 04 against 59 The visualization shows coefficients of the models
for varying C
Out
C100
Sparsity with L1 penalty 625
518 Generalized Linear Models 1183

scikitlearn user guide Release 0213
Sparsity with ElasticNet penalty 469
Sparsity with L2 penalty 469
Score with L1 penalty 090
Score with ElasticNet penalty 090
Score with L2 penalty 090
C010
Sparsity with L1 penalty 2969
Sparsity with ElasticNet penalty 1250
Sparsity with L2 penalty 469
Score with L1 penalty 090
Score with ElasticNet penalty 090
Score with L2 penalty 090
C001
Sparsity with L1 penalty 8438
Sparsity with ElasticNet penalty 6875
Sparsity with L2 penalty 469
Score with L1 penalty 086
Score with ElasticNet penalty 088
Score with L2 penalty 089
printdoc
Authors Alexandre Gramfort alexandregramfortinriafr
Mathieu Blondel mathieublondelorg
Andreas Mueller amuelleraisunibonnde
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import datasets
from sklearn.preprocessing import StandardScaler
digits = datasets.load_digits()
X, y = digits.data, digits.target
X = StandardScaler().fit_transform(X)
classify small against large digits
y = y - 4 * y.astype(np.int)
l1_ratio = 0.5 # L1 weight in the ElasticNet regularization
fig, axes = plt.subplots(3, 3)
Set regularization parameter
for i, C in enumerate(zip(1, 0.1, 0.01)): axes[i, 0] = C
turn down tolerance for short training time
clf1 = LogisticRegression(C=C, penalty='l1', tol=0.01, solver='saga')
clf2 = LogisticRegression(C=C, penalty='l2', tol=0.01, solver='saga')
clf3 = LogisticRegression(C=C, penalty='elasticnet', solver='saga')
l1_ratio = l1_ratio / C
1184 Chapter 5 Examples

```
scikitlearn user guide Release 0213
clf11LRfitX y
clf22LRfitX y
clfenLRfitX y
coef11LR  clf11LRcoefravel
coef22LR  clf22LRcoefravel
coefenLR  clfenLRcoefravel
coef11LR contains zeros due to the
L1 sparsity inducing norm
sparsity11LR  npmeancoef11LR  0 100
sparsity22LR  npmeancoef22LR  0 100
sparsityenLR  npmeancoefenLR  0 100
printC2f  C
print40 2fformatSparsity with L1 penalty sparsity11LR
print40 2fformatSparsity with ElasticNet penalty
sparsityenLR
print40 2fformatSparsity with L2 penalty sparsity22LR
print40 2fformatScore with L1 penalty
clf11LRscoreX y
print40 2fformatScore with ElasticNet penalty
clfenLRscoreX y
print40 2fformatScore with L2 penalty
clf22LRscoreX y
ifi  0
axesrow0settitleL1 penalty
axesrow1settitleElasticNet nl1ratio  s  l1ratio
axesrow2settitleL2 penalty
forax coefs inzipaxesrow coef11LR coefenLR coef22LR
aximshownpabscoefsreshape8 8 interpolationnearest
cmapbinary vmax1 vmin0
axsetxticks
axsetyticks
axesrow0setylabelC  s  C
pltshow
Total running time of the script  0 minutes 0659 seconds
Note Click here to download the full example code
51828 Lasso and Elastic Net
Lasso and elastic net L1 and L2 penalisation implemented using a coordinate descent
The coefficients can be forced to be positive
518 Generalized Linear Models 1185
```


scikitlearn user guide Release 0213

-

518 Generalized Linear Models 1187

scikitlearn user guide Release 0213

•

Out

Computing regularization path using the lasso

Computing regularization path using the positive lasso

Computing regularization path using the elastic net

Computing regularization path using the positive elastic net

printdoc

Author Alexandre Gramfort alexandregramfortinriafr

License BSD (Binary Search Decision) 3 clause

from itertools import cycle

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear_model import lasso_path, enet_path

from sklearn import datasets

diabetes = datasets.load_diabetes

X = diabetes.data

y = diabetes.target

1188 Chapter 5 Examples

```
scikitlearn user guide Release 0213
X Xstdaxis0 Standardize data easier to set the l1ratio parameter
Compute paths
eps 5e3 the smaller it is the longer is the path
printComputing regularization path using the lasso
alphaslasso coefsslasso lassopathX y eps fitinterceptFalse
printComputing regularization path using the positive lasso
alphaspositivelasso coefsspositivelasso lassopath
X y eps positiveTrue fitinterceptFalse
printComputing regularization path using the elastic net
alphasenet coefsenet enetpath
X y epseps l1ratio08 fitinterceptFalse
printComputing regularization path using the positive elastic net
alphaspositiveenet coefsspositiveenet enetpath
X y epseps l1ratio08 positiveTrue fitinterceptFalse
Display results
pltfigure1
colors cycleb r g c k
neglogalphaslasso nplog10alphaslasso
neglogalphasenet nplog10alphasenet
forcoefl coefe c inzipcoefsslasso coefsenet colors
l1 pltplotneglogalphaslasso coefl cc
l2 pltplotneglogalphasenet coefe linestyle cc
pltxlabelLogalpha
pltylabelcoefficients
plttitleLasso and ElasticNet Paths
pltlegendl1 l2 l1 Lasso ElasticNet loclower left
pltaxistight
pltfigure2
neglogalphaspositivelasso nplog10alphaspositivelasso
forcoefl coefpl c inzipcoefsslasso coefsspositivelasso colors
l1 pltplotneglogalphaslasso coefl cc
l2 pltplotneglogalphaspositivelasso coefpl linestyle cc
pltxlabelLogalpha
pltylabelcoefficients
plttitleLasso and positive Lasso
pltlegendl1 l2 l1 Lasso positive Lasso loclower left
pltaxistight
pltfigure3
neglogalphaspositiveenet nplog10alphaspositiveenet
forcoefe coefpe c inzipcoefsenet coefsspositiveenet colors
l1 pltplotneglogalphasenet coefe cc
l2 pltplotneglogalphaspositiveenet coefpe linestyle cc
pltxlabelLogalpha
518 Generalized Linear Models 1189
```

scikitlearn user guide Release 0213

plt.ylabel(coefficients

plt.title(ElasticNet and positive ElasticNet

plt.legend(l1 l2 ElasticNet positive ElasticNet

loc=lower left

plt.axis(tight

plt.show

Total running time of the script 0 minutes 0232 seconds

Note Click here to download the full example code

51829 Automatic Relevance Determination Regression ARD

Fit regression model with Bayesian Ridge Regression

See Bayesian Ridge Regression for more information on the regressor

Compared to the OLS ordinary least squares estimator the coefficient weights are slightly shifted toward zeros

which stabilises them

The histogram of the estimated weights is very peaked as a sparsity inducing prior is implied on the weights

The estimation of the model is done by iteratively maximizing the marginal loglikelihood of the observations

We also plot predictions and uncertainties for ARD for one dimensional regression using polynomial feature expansion

Note the uncertainty starts going up on the right side of the plot This is because these test samples are outside

of the range of the training samples

1190 Chapter 5 Examples

scikitlearn user guide Release 0213

-

518 Generalized Linear Models 1191

scikitlearn user guide Release 0213

-

518 Generalized Linear Models 1193

scikitlearn user guide Release 0213

```
•
printdoc
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
from sklearn.linear_model import ARDRegression, LinearRegression
```

Generating simulated data with Gaussian weights

```
Parameters of the example
nrandomseed0
nsamples nfeatures 100 100
Create Gaussian data
X nrandomrandnnsamples nfeatures
Create weights with a precision lambda of 4
lambda 4
w npzerosnfeatures
Only keep 10 weights of interest
relevantfeatures nrandomrandint0 nfeatures 10
foriinrelevantfeatures
wi statsnormrvsloc0 scale1 npsqrtlambda
Create noise with a precision alpha of 50
alpha 50
1194 Chapter 5 Examples
```


scikitlearn user guide Release 0213
noise statsnormrvsloc0 scale1 npsqrtalpha sizensamples
Create the target
y npdotX w noise

Fit the ARD Regression
clf ARDRegressioncomputescoreTrue
clffitX y
ols LinearRegression
olsfitX y

Plot the true weights the estimated weights the histogram of the
weights and predictions with standard deviations
pltfigurefigsize6 5
plttitleWeights of the model
pltplotclfcoef colordarkblue linestyle linewidth2
labelARD estimate
pltplotolscoef coloryellowgreen linestyle linewidth2
labelOLS estimate
pltplotw colororange linestyle linewidth2 labelGround truth
pltxlabelFeatures
pltylabelValues of the weights
pltlegendloc1
pltfigurefigsize6 5
plttitleHistogram of the weights
pltthistclfcoef binsnfeatures colornavy logTrue
pltscatterclfcoefrelevantfeatures npfullllenrelevantfeatures 5
colorgold markero labelRelevant features
pltylabelFeatures
pltxlabelValues of the weights
pltlegendloc1
pltfigurefigsize6 5
plttitleMarginal loglikelihood
pltplotclfscores colornavy linewidth2
pltylabelScore
pltxlabelIterations
Plotting some predictions for polynomial regression
deffx noiseamount
y npsqrtx npsinx
noise nprandomnormal0 1 lenx
returny noiseamount noise
degree 10
X nplinspace0 10 100
y fX noiseamount1
clfpoly ARDRegressionthresholdlambda1e5
clfpolyfitnpvanderX degree y
Xplot nplinspace0 11 25
yplot fXplot noiseamount0
ymean ystd clfpolypredictnpvanderXplot degree returnstdTrue
518 Generalized Linear Models 1195

scikitlearn user guide Release 0213

pltfigurefigsize6 5

plterrorbarXplot ymean ystd colornavy

labelPolynomial ARD linewidth2

pltplotXplot yplot colorgold linewidth2

labelGround Truth

pltylabelOutput y

pltxlabelFeature X

pltlegendloclower left

pltshow

Total running time of the script 0 minutes 0293 seconds

Note Click here to download the full example code

51830 Bayesian Ridge Regression

Computes a Bayesian Ridge Regression on a synthetic dataset

SeeBayesian Ridge Regression for more information on the regressor

Compared to the OLS ordinary least squares estimator the coefficient weights are slightly shifted toward zeros which stabilises them

As the prior on the weights is a Gaussian prior the histogram of the estimated weights is Gaussian

The estimation of the model is done by iteratively maximizing the marginal loglikelihood of the observations

We also plot predictions and uncertainties for Bayesian Ridge Regression for one dimensional regression using polynomial feature expansion Note the uncertainty starts going up on the right side of the plot This is because these test samples are outside of the range of the training samples

1196 Chapter 5 Examples

scikitlearn user guide Release 0213

-

518 Generalized Linear Models 1197

scikitlearn user guide Release 0213

-

518 Generalized Linear Models 1199

scikitlearn user guide Release 0213

```
•
printdoc
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
from sklearn.linear_model import BayesianRidge, LinearRegression
```

```
Generating simulated data with Gaussian weights
np.random.seed(0)
n_samples, n_features = 100, 100
X = np.random.randn(n_samples, n_features) # Create Gaussian data
# Create weights with a precision lambda of 4
lambda_ = 4
w = np.zeros(n_features)
# Only keep 10 weights of interest
relevant_features = np.random.randint(0, n_features, 10)
for i in relevant_features:
    w[i] = stats.norm.rvs(loc=0, scale=1, npsqrt=lambda_)
# Create noise with a precision alpha of 50
alpha_ = 50
noise = stats.norm.rvs(loc=0, scale=1, npsqrt=alpha_, size=n_samples)
# Create the target
y = np.dot(X, w) + noise
1200 Chapter 5 Examples
```

```
Fit the Bayesian Ridge Regression and an OLS for comparison
clf BayesianRidgecomputescoreTrue
clffitX y
ols LinearRegression
olsfitX y
```

Plot true weights estimated weights histogram of the weights and predictions with standard deviations

```
lw 2
pltfigurefigsize6 5
plttitleWeights of the model
pltplotclfcoef colorlightgreen linewidthlw
labelBayesian Ridge estimate
pltplotw colorgold linewidthlw labelGround truth
pltplotolscoef colornavy linestyle labelOLS estimate
pltxlabelFeatures
pltylabelValues of the weights
pltlegendlocbest propdictsize12
pltfigurefigsize6 5
plttitleHistogram of the weights
plthistclfcoef binsnfeatures colorgold logTrue
edgecolorblack
pltscatterclfcoefrelevantfeatures npfulllenrelevantfeatures 5
colornavy labelRelevant features
pltylabelFeatures
pltxlabelValues of the weights
pltlegendlocupper left
pltfigurefigsize6 5
plttitleMarginal loglikelihood
pltplotclfcores colornavy linewidthlw
pltylabelScore
pltxlabelIterations
```

```
Plotting some predictions for polynomial regression
deffx noiseamount
y npsqrtx npsinx
noise nprandomnormal0 1 lenx
returny noiseamount noise
degree 10
X nplinspace0 10 100
y fX noiseamount01
clfpoly BayesianRidge
clfpolyfitnpvanderX degree y
Xplot nplinspace0 11 25
yplot fXplot noiseamount0
ymean ystd clfpolypredictnpvanderXplot degree returnstdTrue
pltfigurefigsize6 5
plterrorbarXplot ymean ystd colornavy
```

```
scikitlearn user guide Release 0213
labelPolynomial Bayesian Ridge Regression linewidthlw
pltplotXplot yplot colorgold linewidthlw
labelGround Truth
pltlabelOutput y
pltlabelFeature X
pltlegendloclower left
pltshow
```

Total running time of the script 0 minutes 0143 seconds

Note Click here to download the full example code

51831 Lasso model selection CrossValidation AIC BIC

Use the Akaike information criterion AIC the Bayes Information criterion BIC and crossvalidation to select an optimal value of the regularization parameter alpha of the Lasso estimator

Results obtained with LassoLarsIC are based on AICBIC criteria

Informationcriterion based model selection is very fast but it relies on a proper estimation of degrees of freedom are derived for large samples asymptotic results and assume the model is correct ie that the data are actually generated by this model They also tend to break when the problem is badly conditioned more features than samples

For crossvalidation we use 20fold with 2 algorithms to compute the Lasso path coordinate descent as implemented by the LassoCV class and Lars least angle regression as implemented by the LassoLarsCV class Both algorithms give roughly the same results They differ with regards to their execution speed and sources of numerical errors Lars computes a path solution only for each kink in the path As a result it is very efficient when there are only of few kinks which is the case if there are few features or samples Also it is able to compute the full path without setting any meta parameter On the opposite coordinate descent compute the path points on a prespecified grid here we use the default Thus it is more efficient if the number of grid points is smaller than the number of kinks in the path Such a strategy can be interesting if the number of features is really large and there are enough samples to select a large amount In terms of numerical errors for heavily correlated variables Lars will accumulate more errors while the coordinate descent algorithm will only sample the path on a grid

Note how the optimal value of alpha varies for each fold This illustrates why nestedcross validation is necessary when trying to evaluate the performance of a method for which a parameter is chosen by crossvalidation this choice of parameter may not be optimal for unseen data

1202 Chapter 5 Examples

scikitlearn user guide Release 0213

-

518 Generalized Linear Models 1203

scikitlearn user guide Release 0213

•

Out

Computing regularization path using the coordinate descent lasso

Computing regularization path using the Lars lasso

printdoc

Author Olivier Grisel Gael Varoquaux Alexandre Gramfort

License BSD (Binary Search Decision) 3 clause

import time

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear_model import LassoCV LassoLarsCV LassoLarsIC

from sklearn import datasets

This is to avoid division by zero while doing nplog10

EPSILON 1e4

diabetes datasets.load_diabetes

518 Generalized Linear Models 1205

```
scikitlearn user guide Release 0213
X diabetesdata
y diabetestarget
rng nprandomRandomState42
X npcX rngrandomnXshape0 14 add some bad features
normalize data as done by Lars to allow for comparison
X npsqrtnpsumX 2 axis0

LassoLarsIC least angle regression with BICAIC criterion
modelbic LassoLarsICcriterionbic
t1 time
modelbicfitX y
tbic time t1
alphabib modelbicalpha
modelaic LassoLarsICcriterionaic
modelaicfitX y
alphaaic modelaicalpha
defplotbiccriterionmodel name color
alpha modelalpha EPSILON
alphas modelalphas EPSILON
criterion modelcriterion
pltplotnplog10alphas criterion colorcolor
linewidth3 label scriterion name
pltaxvlinenplog10alpha colorcolor linewidth3
labelalpha sestimate name
pltxlabellogalpha
pltylabelcriterion
pltfigure
plotbiccriterionmodelaic AIC b
plotbiccriterionmodelbic BIC r
pltlegend
plttitleInformationcriterion for model selection training time 3fs
tbic

LassoCV coordinate descent
Compute paths
printComputing regularization path using the coordinate descent lasso
t1 time
model LassoCVcv20fitX y
tlassocv time t1
Display results
mlogalphas nplog10modelalphas EPSILON
pltfigure
ymin ymax 2300 3800
pltplotmlogalphas modelmsepath
pltplotmlogalphas modelmsepathmeanaxis1 k
labelAverage across the folds linewidth2
1206 Chapter 5 Examples
```

scikitlearn user guide Release 0213
pltaxvlinenplog10modelalpha EPSILON linestyle color
labelalpha CV (Cross-Validation) estimate
pltlegend
pltxlabellogalpha
pltylabelMean square error
plttitleMean square error on each fold coordinate descent
train time 2fs tlassocv
pltaxistight
pltylimymin ymax

LassoLarsCV least angle regression
Compute paths
printComputing regularization path using the Lars lasso
t1 timetime
model LassoLarsCVcv20fitX y
tlassolarscv timetime t1
Display results
mlogalphas nplog10modelcvalphas EPSILON
pltfigure
pltplotmlogalphas modelmsepath
pltplotmlogalphas modelmsepathmeanaxis1 k
labelAverage across the folds linewidth2
pltaxvlinenplog10modelalpha linestyle color
labelalpha CV (Cross-Validation)
pltlegend
pltxlabellogalpha
pltylabelMean square error
plttitleMean square error on each fold Lars train time 2fs
tlassolarscv
pltaxistight
pltylimymin ymax
pltshow

Total running time of the script 0 minutes 0811 seconds
Note Click here to download the full example code
51832 Multiclass sparse logisitic regression on newgroups20
Comparison of multinomial logistic L1 vs oneversusrest L1 logistic regression to classify documents from the new
groups20 dataset Multinomial logistic regression yields more accurate results and is faster to train on the larger scale
dataset
Here we use the l1 sparsity that trims the weights of not informative features to zero This is good if the goal is to
extract the strongly discriminative vocabulary of each class If the goal is to get the best predictive accuracy it is better
to use the non sparsityinducing l2 penalty instead
518 Generalized Linear Models 1207

scikitlearn user guide Release 0213

A more traditional and possibly better way to predict on a sparse subset of input features would be to use univariate feature selection followed by a traditional l2penalised logistic regression model

Out

Dataset 20newsgroup trainsamples9000 nfeatures130107 nclasses20

modelOne versus Rest solversaga Number of epochs 1

modelOne versus Rest solversaga Number of epochs 2

modelOne versus Rest solversaga Number of epochs 4

Test accuracy for model ovr 07490

nonzero coefficients for model ovr per class

031743104 036815852 04181174 046115889 024595141 041350581

031281945 027054655 058720899 032972861 04158116 03312658

041888599 041120001 059643217 031666244 034279478 028130692

035278655 024748861

Run time 4 epochs for model ovr306

modelMultinomial solversaga Number of epochs 1

modelMultinomial solversaga Number of epochs 3

modelMultinomial solversaga Number of epochs 7

Test accuracy for model multinomial 07450

nonzero coefficients for model multinomial per class

013219888 011452112 013066169 013681047 012066991 015909982

013450468 009146318 007916561 012143851 013911627 010760374

018984374 012143851 017524038 022289346 011605832 007916561

007301682 015141384

Run time 7 epochs for model multinomial255

Example run in 11262 s

1208 Chapter 5 Examples

```
scikitlearn user guide Release 0213
import timeit
import warnings
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import fetch20newsgroupsvectorized
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.exceptions import ConvergenceWarning
printdoc
    Author Arthur Mensch
warnings.filterwarnings(ignore, category=ConvergenceWarning)
modulesklearn
t0 = timeit.default_timer
    We use SAGA solver
solver = 'saga'
    Turn down for faster run time
nsamples = 10000
    Memorized fetchrcv1 for faster access
dataset = fetch20newsgroupsvectorizedall
X = dataset.data
y = dataset.target
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    random_state=42,
                                                    stratify=y,
                                                    test_size=0.1,
                                                    train_samples=nfeatures,
                                                    X_train_shape=X_train.shape,
                                                    n_classes=np.unique(y).shape[0])
printDataset(20newsgroup, train_samples, nfeatures, n_classes,
              train_samples, nfeatures, n_classes)
models = {'ovr': {'name': 'One versus Rest', 'iters': [1, 2, 4]},
          'multinomial': {'name': 'Multinomial', 'iters': [1, 3, 7]}}
formodelinmodels
    Add initial chance level values for plotting purpose
accuracies = {}
times = {}
densities = {}
modelparams = {}
modelsmodel
518 Generalized Linear Models 1209
```

```
scikitlearn user guide Release 0213
Small number of epochs for fast runtime
forthismaxiter inmodelparamsiters
printmodel s solver s Number of epochs s
modelparamsname solver thismaxiter
lr LogisticRegressionsolversolver
multiclassmodel
C1
penaltyl1
fitinterceptTrue
maxiterthismaxiter
randomstate42

t1 timeitdefaulttimer
lrfitXtrain ytrain
traintime timeitdefaulttimer t1
ypred lrpredictXtest
accuracy npsumypred ytest ytestshape0
density npmeanlrcoef 0 axis1 100
accuraciesappendaccuracy
densitiesappenddensity
timesappendtraintime
modelsmodeltimes times
modelsmodeldensities densities
modelsmodelaccuracies accuracies
printTest accuracy for model s4f model accuracies1
printnonzero coefficients for model s
per class ns model densities1
printRun time iepochs for model s
2f modelparamsiters1 model times1
fig pltfigure
ax figaddsubplot111
formodelinmodels
name modelsmodelname
times modelsmodeltimes
accuracies modelsmodelaccuracies
axplottimes accuracies markero
labelModel s name
axsetxlabelTrain time s
axsetylabelTest accuracy
axlegend
figsuptitleMultinomial vs OnevsRest Logistic L1 n
Dataset s 20newsgroups
figtightlayout
figsubplotsadjusttop085
runtime timeitdefaulttimer t0
printExample run in 3fs runtime
pltshow
Total running time of the script 0 minutes 11263 seconds
Note Click here to download the full example code
1210 Chapter 5 Examples
```


scikitlearn user guide Release 0213

51833 Early stopping of Stochastic Gradient Descent

Stochastic Gradient Descent is an optimization technique which minimizes a loss function in a stochastic fashion performing a gradient descent step sample by sample. In particular, it is a very efficient method to fit linear models. As a stochastic method, the loss function is not necessarily decreasing at each iteration and convergence is only guaranteed in expectation. For this reason, monitoring the convergence on the loss function can be difficult. Another approach is to monitor convergence on a validation score. In this case, the input data is split into a training set and a validation set. The model is then fitted on the training set and the stopping criterion is based on the prediction score computed on the validation set. This enables us to find the least number of iterations which is sufficient to build a model that generalizes well to unseen data and reduces the chance of overfitting the training data. This early stopping strategy is activated if `earlystopping=True`; otherwise, the stopping criterion only uses the training loss on the entire input data. To better control the early stopping strategy, we can specify a parameter `validationfraction` which sets the fraction of the input dataset that we keep aside to compute the validation score. The optimization will continue until the validation score did not improve by at least `tol` during the last `niter` iterations. The actual number of iterations is available at the attribute `niter`. This example illustrates how the early stopping can be used in the `sklearn.linear_model.SGDClassifier` model to achieve almost the same accuracy as compared to a model built without early stopping. This can significantly reduce training time. Note that scores differ between the stopping criteria even from early iterations because some of the training data is held out with the validation stopping criterion.

Out

No stopping criterion

Training loss

Validation score

518 Generalized Linear Models 1211

scikitlearn user guide Release 0213
Authors Tom Dupre la Tour

```
License BSD (Binary Search Decision) 3 clause
import time
import sys
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linearmodel
from sklearn.datasets import fetchopenml
from sklearn.model_selection import train_test_split
from sklearn.utils.testing import ignore_warnings
from sklearn.exceptions import ConvergenceWarning
from sklearn.utils import shuffle
printdoc
def load_mnist_data(n_samples=None, class0=0, class1=1):
    """Load MNIST select two classes shuffle and return only n_samples
    Load data from http://openml.org/d/554
    mnist = fetch_openml('mnist_784', version=1)
    take only two classes for binary classification
    mask = np.logical_or(mnist.target == class0, mnist.target == class1)
    X, y = shuffle(mnist.data[mask], mnist.target[mask], random_state=42)
    if n_samples is not None:
        X, y = X[:n_samples], y[:n_samples]
    return X, y
ignore_warnings(category=ConvergenceWarning)
def fit_and_score(estimator, X_train, X_test, y_train, y_test):
    """Fit the estimator on the train set and score it on both sets
    estimator.set_params(max_iter=max_iter)
    estimator.set_params(random_state=0)
    start = time.time()
    estimator.fit(X_train, y_train)
    fit_time = time.time() - start
    n_iter = estimator.n_iter_
    train_score = estimator.score(X_train, y_train)
    test_score = estimator.score(X_test, y_test)
    return fit_time, n_iter, train_score, test_score
    Define the estimators to compare
    estimator_dict = {}
    No stopping criterion
    linear_model.SGDClassifier(tol=1e-3, n_iter_no_change=3)
1212 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Training loss
linearmodelSGDClassifierearlystoppingFalse niternochange3
tol01
Validation score
linearmodelSGDClassifierearlystoppingTrue niternochange3
tol00001 validationfraction02

Load the dataset
X y loadmnistnsamples10000
Xtrain Xtest ytrain ytest traintestsplitX y testsize05
randomstate0
results
forestimatorname estimator inestimatordictitems
printestimatorname end
formaxiter inrange1 50
print end
sysstdoutflush
fittime niter trainscore testscore fitandscore
estimator maxiter Xtrain Xtest ytrain ytest
resultsappendestimatorname maxiter fittime niter
trainscore testscore
print
Transform the results in a pandas dataframe for easy plotting
columns
Stopping criterion maxiter Fit time sec niter
Train score Test score

resultsdf pdDataFramerresults columnscolumns
Define what to plot xaxis yaxis
lines Stopping criterion
plotlist
maxiter Train score
maxiter Test score
maxiter niter
maxiter Fit time sec

nrows 2
ncols intnpceillenplotlist 2
fig axes pltsubplotsnrowsnrows ncolsncols figsize6 nc
4nrows
axes0 0getsharedyaxesjoinaxes0 0 axes0 1
forax xaxis yaxis inzipaxesravel plotlist
forcriterion groupdf inresultsdfgroupbylines
groupdfplotxxaxis yyaxis labelcriterion axax
axsettityaxis
axlegendtitlelines
figtightlayout
pltshow
518 Generalized Linear Models 1213

scikitlearn user guide Release 0213
Total running time of the script 0 minutes 45461 seconds
519 Manifold learning
Examples concerning the sklearnmanifold module
Note [Click here to download the full example code](#)
5191 Swiss Roll reduction with LLE
An illustration of Swiss Roll reduction with locally linear embedding
Out
Computing LLE embedding
Done Reconstruction error 732714e08
1214 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Author Fabian Pedregosa fabianpedregosainriafr
License BSD (Binary Search Decision) 3 clause C INRIA 2011
printdoc
import matplotlib.pyplot as plt
This import is needed to modify the way figure behaves
from mpltoolkitsmplot3d import Axes3D
Axes3D

Locally linear embedding of the swiss roll
from sklearn import manifold datasets
X color datasetssamplesgeneratormakeswissrollnsamples1500
printComputing LLE embedding
Xr err manifoldlocallylinearembddingX nneighbors12
ncomponents2
printDone Reconstruction error g err

Plot result
fig pltfigure
ax figaddsubplot211 projection3d
axscatterX 0 X 1 X 2 ccolor cmappltcmSpectral
axsettitleOriginal data
ax figaddsubplot212
axscatterXr 0 Xr 1 ccolor cmappltcmSpectral
pltaxistight
pltxticks pltxticks
plttitleProjected data
pltshow
Total running time of the script 0 minutes 0182 seconds
Note Click here to download the full example code
5192 Multidimensional scaling
An illustration of the metric and nonmetric MDS on generated noisy data
The reconstructed points using the metric MDS and non metric MDS are slightly shifted to avoid overlapping
519 Manifold learning 1215
```

```
scikitlearn user guide Release 0213
Author Nelle Varoquaux nellevaroquauxgmailcom
License BSD (Binary Search Decision)
printdoc
import numpy as np
from matplotlib import pyplotasplt
from matplotlibcollections import LineCollection
from sklearn import manifold
from sklearnmetrics import euclideandistances
from sklearndecomposition import PCA (Principal Component Analysis)
nsamples 20
seed np.random.RandomState(seed3)
Xtrue seedrandint(0, 20, 2, nsamples, dtype=np.float)
Xtrue Xtrue.reshape(nsamples, 2)
Center the data
Xtrue Xtrue.mean()
similarities euclideandistances(Xtrue)
Add noise to the similarities
noise np.random.randn(nsamples, nsamples)
noise noise noiseT
noisenparangenoiseshape(0, nparangenoisesshape(0) - 0)
1216 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
similarities noise
mds manifoldMDSncomponents2 maxiter3000 eps1e9 randomstateseed
dissimilarityprecomputed njobs1
pos mdsfitsimilaritiesembedding
nmbs manifoldMDSncomponents2 metricFalse maxiter3000 eps1e12
dissimilarityprecomputed randomstateseed njobs1
ninit1
npos nmbsfittransforms similarities initpos
Rescale the data
pos npsqrtXtrue 2sum npsqrtpos 2sum
npos npsqrtXtrue 2sum npsqrtnpos 2sum
Rotate the data
clf PCAncomponents2
Xtrue clffittransformXtrue
pos clffittransformpos
npos clffittransformnpos
fig pltfigure1
ax pltaxes0 0 1 1
s 100
pltscatterXtrue 0 Xtrue 1 colornavy ss lw0
labelTrue Position
pltscatterpos 0 pos 1 colorturquoise ss lw0 labelMDS
pltscatternpos 0 npos 1 colordarkorange ss lw0 labelNMDS
pltlegendscatterpoints1 locbest shadowFalse
similarities similaritiesmax similarities 100
similaritiesnpisinfsimilarities 0
Plot the edges
startidx endidx npwherepos
a sequence of line0line1line2 where
linen x0 y0 x1 y1 xm ym
segments Xtruei Xtruej
foriinrangelenpos forjinrangelenpos
values npabssimilarities
lc LineCollectionsegments
zorder0 cmappltcmBlues
normpltNormalize0 valuesmax
lcsetarraysimilaritiesflatten
lcsetlinewidthsnpfulllensegments 05
axaddcollectionlc
pltshow
Total running time of the script 0 minutes 0063 seconds
Note Click here to download the full example code
519 Manifold learning 1217
```

scikitlearn user guide Release 0213

5193 tSNE The effect of various perplexity values on the shape

An illustration of tSNE on the two concentric circles and the Scurve datasets for different perplexity values

We observe a tendency towards clearer shapes as the perplexity value increases

The size the distance and the shape of clusters may vary upon initialization perplexity values and does not always convey a meaning

As shown below tSNE for higher perplexities finds meaningful topology of two concentric circles however the size and the distance of the circles varies slightly from the original Contrary to the two circles dataset the shapes visually diverge from Scurve topology on the Scurve dataset even for larger perplexity values

For further details “How to Use tSNE Effectively” <https://distill.pub/2016/misreadtsne> provides a good discussion of the effects of various parameters as well as interactive plots to explore those effects

Out

circles perplexity5 in 089 sec

circles perplexity30 in 12 sec

circles perplexity50 in 13 sec

circles perplexity100 in 18 sec

Scurve perplexity5 in 092 sec

Scurve perplexity30 in 12 sec

Scurve perplexity50 in 14 sec

Scurve perplexity100 in 19 sec

uniform grid perplexity5 in 088 sec

uniform grid perplexity30 in 11 sec

uniform grid perplexity50 in 11 sec

uniform grid perplexity100 in 17 sec

1218 Chapter 5 Examples


```
scikitlearn user guide Release 0213
Author Narine Kokhlikyan narineslice.com
License BSD (Binary Search Decision)
printdoc
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import NullFormatter
from sklearn import manifold, datasets
from time import time

nsamples = 300
ncomponents = 2
fig, subplots = plt.subplots(3, 5, figsize=(15, 8))
perplexities = [5, 30, 50, 100]
X, y = datasets.make_circles(nsamples, nsamples, factor=5, noise=0.5)
red = y == 0
green = y == 1
ax = subplots[0, 0]
ax.scatter(X[red], X[green], c='cr')
ax.scatter(X[green], X[green], c='cg')
ax.xaxis.set_major_formatter(NullFormatter)
ax.yaxis.set_major_formatter(NullFormatter)
plt.axis('tight')
for i, perplexity in enumerate(perplexities):
    ax = subplots[0, i]
    t0 = time()
    tsne = manifold.TSNE(ncomponents, ncomponents, init='random',
        random_state=0, perplexity=perplexity)
    Y = tsne.fit_transform(X)
    t1 = time()
    print('circles: perplexity %d in %2gsec' % (perplexity, t1 - t0))
    ax.set_title('Perplexity %d' % perplexity)
    ax.scatter(Y[red], Y[green], c='cr')
    ax.scatter(Y[green], Y[green], c='cg')
    ax.xaxis.set_major_formatter(NullFormatter)
    ax.yaxis.set_major_formatter(NullFormatter)
    ax.axis('tight')

# Another example using scurve
X, color = datasets.samples_generator.make_s_curve(nsamples, random_state=0)
ax = subplots[1, 0]
ax.scatter(X, X, c=color)
ax.xaxis.set_major_formatter(NullFormatter)
ax.yaxis.set_major_formatter(NullFormatter)
for i, perplexity in enumerate(perplexities):
    ax = subplots[1, i]
    t0 = time()
    tsne = manifold.TSNE(ncomponents, ncomponents, init='random',
        random_state=0, perplexity=perplexity)
    Y = tsne.fit_transform(X)
    t1 = time()
    print('scurve: perplexity %d in %2gsec' % (perplexity, t1 - t0))
    ax.set_title('Perplexity %d' % perplexity)
    ax.scatter(Y, Y, c='cr')
    ax.scatter(Y[green], Y[green], c='cg')
    ax.xaxis.set_major_formatter(NullFormatter)
    ax.yaxis.set_major_formatter(NullFormatter)
    ax.axis('tight')
```

scikitlearn user guide Release 0213

```
randomstate0 perplexityperplexity
Y tsnefittransformX
t1 time
printScurve perplexity din2gsec perplexity t1 t0
axsettitlePerplexity d perplexity
axscatterY 0 Y 1 ccolor
axxaxissetmajorformatterNullFormatter
axyaxissetmajorformatterNullFormatter
axaxistight
Another example using a 2D uniform grid
x nplinspace0 1 intnpsqrtnsamples
xx yy npmeshgridx x
X nphstack
xxravelreshape1 1
yyravelreshape1 1
```

```
color xxravel
ax subplots20
axscatterX 0 X 1 ccolor
axxaxissetmajorformatterNullFormatter
axyaxissetmajorformatterNullFormatter
fori perplexity inenumerateperplexities
ax subplots2i 1
t0 time
tsne manifoldTSNEcomponentsncomponents initrandom
randomstate0 perplexityperplexity
Y tsnefittransformX
t1 time
printuniform grid perplexity din2gsec perplexity t1 t0
axsettitlePerplexity d perplexity
axscatterY 0 Y 1 ccolor
axxaxissetmajorformatterNullFormatter
axyaxissetmajorformatterNullFormatter
axaxistight
pltshow
```

Total running time of the script 0 minutes 15568 seconds

Note Click here to download the full example code

5194 Comparison of Manifold Learning methods

An illustration of dimensionality reduction on the Scurve dataset with various manifold learning methods

For a discussion and comparison of these algorithms see the manifold module page

For a similar example where the methods are applied to a sphere dataset see Manifold Learning methods on a severed sphere

1220 Chapter 5 Examples

scikitlearn user guide Release 0213

Note that the purpose of the MDS is to find a lowdimensional representation of the data here 2D in which the distances respect well the distances in the original highdimensional space unlike other manifoldlearning algorithms it does not seeks an isotropic representation of the data in the lowdimensional space

```
Out
standard 0095 sec
Itsa 02 sec
hessian 036 sec
modified 02 sec
Isomap 036 sec
MDS 2 sec
SpectralEmbedding 01 sec
tSNE 6 sec
  Author Jake Vanderplas  vanderplasastrowashingtonedu
printdoc
from time import time
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.ticker import NullFormatter
from sklearn import manifold datasets
  Next line to silence pyflakes This import is needed
Axes3D
519 Manifold learning 1221
```

```
scikitlearn user guide Release 0213
npoints 1000
X color datasetssamplesgeneratormakescurvenpoints randomstate0
nneighbors 10
ncomponents 2
fig pltfigurefigsize15 8
pltsuptitleManifold Learning with ipoints ineighbors
1000 nneighbors fontsize14
ax figaddsubplot251 projection3d
axscatterX 0 X 1 X 2 ccolor cmappltcmSpectral
axviewinit4 72
methods standard ltsa hessian modified
labels LLE LTSA Hessian LLE Modified LLE
fori method inenumeratemethods
t0 time
Y manifoldLocallyLinearEmbeddingnneighbors ncomponents
eigensolverauto
methodmethodfittransformX
t1 time
prints2gsec methodsi t1 t0
ax figaddsubplot252 i
pltscatterY 0 Y 1 ccolor cmappltcmSpectral
plttitle s2gsec labelsi t1 t0
axxaxissetmajorformatterNullFormatter
axyaxissetmajorformatterNullFormatter
pltaxistight
t0 time
Y manifoldIsomapnneighbors ncomponentsfittransformX
t1 time
printIsomap 2gsec t1 t0
ax figaddsubplot257
pltscatterY 0 Y 1 ccolor cmappltcmSpectral
plttitleIsomap 2gsec t1 t0
axxaxissetmajorformatterNullFormatter
axyaxissetmajorformatterNullFormatter
pltaxistight
t0 time
mds manifoldMDSncomponents maxiter100 ninit1
Y mdsfittransformX
t1 time
printMDS2gsec t1 t0
ax figaddsubplot258
pltscatterY 0 Y 1 ccolor cmappltcmSpectral
plttitleMDS 2gsec t1 t0
axxaxissetmajorformatterNullFormatter
axyaxissetmajorformatterNullFormatter
pltaxistight
t0 time
1222 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
se manifoldSpectralEmbeddingncomponentsncomponents
nneighborsnneighbors
Y sefittransformX
t1 time
printSpectralEmbedding 2gsec t1 t0
ax figaddsubplot259
pltscatterY 0 Y 1 ccolor cmappltcmSpectral
plttitleSpectralEmbedding 2gsec t1 t0
axxaxisetmajorformatterNullFormatter
axyaxisetmajorformatterNullFormatter
pltaxistight
t0 time
tsne manifoldTSNEcomponentsncomponents initpca randomstate0
Y tsnefittransformX
t1 time
printtSNE 2gsec t1 t0
ax figaddsubplot2 5 10
pltscatterY 0 Y 1 ccolor cmappltcmSpectral
plttitletSNE 2gsec t1 t0
axxaxisetmajorformatterNullFormatter
axyaxisetmajorformatterNullFormatter
pltaxistight
pltshow
```

Total running time of the script 0 minutes 9515 seconds

Note [Click here to download the full example code](#)

5195 Manifold Learning methods on a severed sphere

An application of the different Manifold learning techniques on a spherical dataset Here one can see the use of dimensionality reduction in order to gain some intuition regarding the manifold learning methods Regarding the dataset the poles are cut from the sphere as well as a thin slice down its side This enables the manifold learning techniques to ‘spread it open’ whilst projecting it onto two dimensions

For a similar example where the methods are applied to the Scurve dataset see [Comparison of Manifold Learning methods](#)

Note that the purpose of the MDS is to find a lowdimensional representation of the data here 2D in which the distances respect well the distances in the original highdimensional space unlike other manifoldlearning algorithms it does not seeks an isotropic representation of the data in the lowdimensional space Here the manifold problem matches fairly that of representing a flat map of the Earth as with map projection

519 Manifold learning 1223

scikitlearn user guide Release 0213
Out
standard 0069 sec
Itsa 012 sec
hessian 029 sec
modified 016 sec
ISO 027 sec
MDS 12 sec
Spectral Embedding 015 sec
tSNE 35 sec
 Author Jaques Grobler jaquesgroblerinriafr
 License BSD (Binary Search Decision) 3 clause
 printdoc
 from time import time
 import numpy as np
 import matplotlib.pyplot as plt
 from mpl_toolkits.mplot3d import Axes3D
 from matplotlib.ticker import NullFormatter
 from sklearn import manifold
 from sklearn.utils import check_random_state
 Next line to silence pyflakes
 Axes3D
 Variables for manifold learning
1224 Chapter 5 Examples

```
scikitlearn user guide Release 0213
nneighbors 10
nsamples 1000
    Create our sphere
randomstate checkrandomstate0
p randomstaterandnsamples 2nppi 055
t randomstaterandnsamples nppi
    Sever the poles from the sphere
indices t nppi nppi 8 t nppi 8
colors pindices
x y z npsintindices npcospindices
npsintindices npsinindices
npcostindices
    Plot our dataset
fig pltfigurefigsize15 8
pltsuptitleManifold Learning with ipoints ineighbors
    1000 nneighbors fontsize14
ax figaddsubplot251 projection3d
axscatterx y z cpindices cmappltcmrainbow
axviewinit40 10
spheredata nparrayx y zT
    Perform Locally Linear Embedding Manifold learning
methods standard ltsa hessian modified
labels LLE LTSA Hessian LLE Modified LLE
fori method inenumeratemethods
t0 time
transdata manifold
LocallyLinearEmbeddingnneighbors 2
methodmethodfittransformspgheredataT
t1 time
prints2gsec methodsi t1 t0
ax figaddsubplot252 i
pltscattertransdata0 transdata1 ccolors cmappltcmrainbow
plttitle s2gsec labelsi t1 t0
axxaxisetmajorformatterNullFormatter
axyaxisetmajorformatterNullFormatter
pltaxistight
    Perform Isomap Manifold learning
t0 time
transdata manifoldIsomapnneighbors ncomponents2
fittransformspgheredataT
t1 time
prints2gsec ISO t1 t0
ax figaddsubplot257
pltscattertransdata0 transdata1 ccolors cmappltcmrainbow
plttitle s2gsec Isomap t1 t0
axxaxisetmajorformatterNullFormatter
axyaxisetmajorformatterNullFormatter
pltaxistight
519 Manifold learning 1225
```

```
scikitlearn user guide Release 0213
Perform Multidimensional scaling
t0 time
mds manifoldMDS2 maxiter100 ninit1
transdata mdsfittransformspheredataT
t1 time
printMDS2gsec t1 t0
ax figaddsubplot258
pltscattertransdata0 transdata1 ccolors cmappltcmrainbow
plttitleMDS 2gsec t1 t0
axxaxisetmajorformatterNullFormatter
axyaxisetmajorformatterNullFormatter
pltaxistight
Perform Spectral Embedding
t0 time
se manifoldSpectralEmbeddingncomponents2
nneighborsneighbors
transdata sefittransformspheredataT
t1 time
printSpectral Embedding 2gsec t1 t0
ax figaddsubplot259
pltscattertransdata0 transdata1 ccolors cmappltcmrainbow
plttitleSpectral Embedding 2gsec t1 t0
axxaxisetmajorformatterNullFormatter
axyaxisetmajorformatterNullFormatter
pltaxistight
Perform tdistributed stochastic neighbor embedding
t0 time
tsne manifoldTSNEcomponents2 initpca randomstate0
transdata tsnefittransformspheredataT
t1 time
printtSNE 2gsec t1 t0
ax figaddsubplot2 5 10
pltscattertransdata0 transdata1 ccolors cmappltcmrainbow
plttitletSNE 2gsec t1 t0
axxaxisetmajorformatterNullFormatter
axyaxisetmajorformatterNullFormatter
pltaxistight
pltshow
Total running time of the script 0 minutes 5954 seconds
Note Click here to download the full example code
5196 Manifold learning on handwritten digits Locally Linear Embedding
Isomap
An illustration of various embeddings on the digits dataset
1226 Chapter 5 Examples
```


scikitlearn user guide Release 0213

The RandomTreesEmbedding from the sklearnensemble module is not technically a manifold embedding method as it learn a highdimensional representation on which we apply a dimensionality reduction method However it is often useful to cast a dataset into a representation in which the classes are linearlyseparable

tSNE will be initialized with the embedding that is generated by PCA (Principal Component Analysis) in this example which is not the default setting

It ensures global stability of the embedding ie the embedding does not depend on random initialization

Linear Discriminant Analysis from the sklearniscriminantanalysis module and Neighborhood Components Analysis from the sklearnneighbors module are supervised dimensionality reduction method ie they make use of the provided labels contrary to other methods

-

519 Manifold learning 1227

scikitlearn user guide Release 0213

-

519 Manifold learning 1229

scikitlearn user guide Release 0213

-

519 Manifold learning 1231

scikitlearn user guide Release 0213

- 519 Manifold learning 1233

scikitlearn user guide Release 0213

- 519 Manifold learning 1235

scikitlearn user guide Release 0213

- 519 Manifold learning 1237

scikitlearn user guide Release 0213

- 519 Manifold learning 1239

scikitlearn user guide Release 0213

•

Out

Computing random projection

Computing PCA (Principal Component Analysis) projection

Computing Linear Discriminant Analysis projection

Computing Isomap projection

Done

Computing LLE embedding

Done Reconstruction error 163544e06

Computing modified LLE embedding

Done Reconstruction error 0360652

Computing Hessian LLE embedding

Done Reconstruction error 0212801

Computing LTSA embedding

Done Reconstruction error 0212808

Computing MDS embedding

Done Stress 148085982692961

Computing Totally Random Trees embedding

Computing Spectral embedding

Computing tSNE embedding

Computing NCA projection

1240 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Authors Fabian Pedregosa fabianpedregosainriafr
Olivier Grisel oliviergrisenstaorg
Mathieu Blondel mathieumblondelorg
Gael Varoquaux
License BSD (Binary Search Decision) 3 clause C INRIA 2011
printdoc
from time import time
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import offsetbox
from sklearn import manifold datasets decomposition ensemble
discriminantanalysis randomprojection neighbors
digits datasetsloadigitsnclass6
X digitsdata
y digitstarget
nsamples nfeatures Xshape
nneighbors 30

Scale and visualize the embedding vectors
defplotembeddingX titleNone
xmin xmax npminX 0 npmaxX 0
X X xmin xmax xmin
pltfigure
ax pltsubplot111
foriinrangeXshape0
plttextXi 0 Xi 1 stryi
colorpltcmSet1yi 10
fontdictweight bold size 9
ifhasattroffsetbox AnnotationBbox
only print thumbnails with matplotlib 10
shownimages nparray1 1 just something big
foriinrangeXshape0
dist npsumXi shownimages 2 1
ifnpsmindist 4e3
dont show points that are too close
continue
shownimages nprshownimages Xi
imagebox offsetboxAnnotationBbox
offsetboxOffsetImagedigitsimagesi cmappltcmgrayr
Xi
axaddartistimagebox
pltxticks pltxticks
iftitleis notNone
plttitletitle

Plot images of the digits
nimgperrow 20
img npzeros10 nimgperrow 10 nimgperrow
foriinrangenimgperrow
519 Manifold learning 1241
```

scikitlearn user guide Release 0213

```
ix 10 i 1
forjinrangenimgperrow
iy 10 j 1
imgixix 8 iyiy 8 Xi nimgperrow jreshape8 8
pltimshowimg cmappltcmbinary
pltxticks
plttyticks
plttitleA selection from the 64dimensional digits dataset
```

```
Random 2D projection using a random unitary matrix
printComputing random projection
rp randomprojectionSparseRandomProjectionncomponents2 randomstate42
Xprojected rpfitttransformX
plotembeddingXprojected Random Projection of the digits
```

```
Projection on to the first 2 principal components
printComputing PCA (Principal Component Analysis) projection
t0 time
Xpca decompositionTruncatedSVDncomponents2fitttransformX
plotembeddingXpca
Principal Components projection of the digits time 2fs
time t0
```

```
Projection on to the first 2 linear discriminant components
printComputing Linear Discriminant Analysis projection
X2 Xcopy
X2flatXshape1 1 001 Make X invertible
t0 time
Xlda discriminantanalysisLinearDiscriminantAnalysisncomponents2fit
↔transformX2 y
plotembeddingXlda
Linear Discriminant projection of the digits time 2fs
time t0
```

```
Isomap projection of the digits dataset
printComputing Isomap projection
t0 time
Xiso manifoldIsomapneighbors ncomponents2fitttransformX
printDone
plotembeddingXiso
Isomap projection of the digits time 2fs
time t0
```

```
Locally linear embedding of the digits dataset
printComputing LLE embedding
clf manifoldLocallyLinearEmbeddingneighbors ncomponents2
1242 Chapter 5 Examples
```


scikitlearn user guide Release 0213

methodstandard

t0 time

Xlle clffittransformX

printDone Reconstruction error g clfreconstructionerror

plotembeddingXlle

Locally Linear Embedding of the digits time 2fs

time t0

Modified Locally linear embedding of the digits dataset

printComputing modified LLE embedding

clf manifoldLocallyLinearEmbeddingnneighbors ncomponents2

methodmodified

t0 time

Xmlle clffittransformX

printDone Reconstruction error g clfreconstructionerror

plotembeddingXmlle

Modified Locally Linear Embedding of the digits time 2fs

time t0

HLL embedding of the digits dataset

printComputing Hessian LLE embedding

clf manifoldLocallyLinearEmbeddingnneighbors ncomponents2

methodhessian

t0 time

Xhlle clffittransformX

printDone Reconstruction error g clfreconstructionerror

plotembeddingXhlle

Hessian Locally Linear Embedding of the digits time 2fs

time t0

LTSA embedding of the digits dataset

printComputing LTSA embedding

clf manifoldLocallyLinearEmbeddingnneighbors ncomponents2

methodltsa

t0 time

Xltsa clffittransformX

printDone Reconstruction error g clfreconstructionerror

plotembeddingXltsa

Local Tangent Space Alignment of the digits time 2fs

time t0

MDS embedding of the digits dataset

printComputing MDS embedding

clf manifoldMDSncomponents2 ninit1 maxiter100

t0 time

Xmds clffittransformX

printDone Stress f clfstress

plotembeddingXmds

MDS embedding of the digits time 2fs

time t0

519 Manifold learning 1243

scikitlearn user guide Release 0213

Random Trees embedding of the digits dataset
printComputing Totally Random Trees embedding
hasher ensembleRandomTreesEmbeddingnestimators200 randomstate0
maxdepth5
t0 time
Xtransformed hasherfittransformX
pca decompositionTruncatedSVDncomponents2
Xreduced pcafittransformXtransformed
plotembeddingXreduced
Random forest embedding of the digits time 2fs
time t0

Spectral embedding of the digits dataset
printComputing Spectral embedding
embedder manifoldSpectralEmbeddingncomponents2 randomstate0
eigensolverarpack
t0 time
Xse embedderfittransformX
plotembeddingXse
Spectral embedding of the digits time 2fs
time t0

tSNE embedding of the digits dataset
printComputing tSNE embedding
tsne manifoldTSNEencomponents2 initpca randomstate0
t0 time
Xtsne tsnefittransformX
plotembeddingXtsne
tSNE embedding of the digits time 2fs
time t0

NCA projection of the digits dataset
printComputing NCA projection
nca neighborsNeighborhoodComponentsAnalysisncomponents2 randomstate0
t0 time
Xnca ncafittransformX y
plotembeddingXnca
NCA embedding of the digits time 2fs
time t0
pltshow
Total running time of the script 0 minutes 17641 seconds
520 Gaussian Mixture Models
Examples concerning the sklearnmixture module
1244 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Note Click here to download the full example code
5201 Density Estimation for a Gaussian mixture
Plot the density estimation of a mixture of two Gaussians Data is generated from two Gaussians with different centers
and covariance matrices
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import LogNorm
from sklearn import mixture
nsamples = 300
generate random sample two components
np.random.seed(0)
generate spherical data centered on 20 20
shiftedgaussian = np.random.randn(nsamples, 2) + np.array([20, 20])
generate zero centered stretched Gaussian data
C = np.array([0.07, 35, 7])
stretchedgaussian = np.dot(np.random.randn(nsamples, 2), C)
520 Gaussian Mixture Models 1245
```

scikitlearn user guide Release 0213

```
concatenate the two datasets into the final training set
Xtrain np.vstack(shiftedgaussian, stretchedgaussian)
fit a Gaussian Mixture Model with two components
clf = mixture.GaussianMixture(n_components=2, covariance_type='full')
clf.fit(Xtrain)
display predicted scores by the model as a contour plot
x = np.linspace(20, 30)
y = np.linspace(20, 40)
X, Y = np.meshgrid(x, y)
XX = np.array(X.ravel(), Y.ravel())
Z = clf.score_samples(XX)
Z = Z.reshape(X.shape)
CS = plt.contour(X, Y, Z, norm=LogNorm, vmin=10, vmax=10000)
levels = np.logspace(0, 3, 10)
CB = plt.colorbar(CS, shrink=0.8, extend=both)
plt.scatter(Xtrain, 0, Xtrain, 1, 8)
plt.title('Negative loglikelihood predicted by a GMM')
plt.axis('tight')
plt.show()
Total running time of the script: 0 minutes 00.40 seconds
Note: Click here to download the full example code
5202 Gaussian Mixture Model Ellipsoids
Plot the confidence ellipsoids of a mixture of two Gaussians obtained with Expectation Maximisation
GaussianMixture class and Variational Inference BayesianGaussianMixture class models with a
Dirichlet process prior
Both models have access to five components with which to fit the data. Note that the Expectation Maximisation
model will necessarily use all five components while the Variational Inference model will effectively only use as many
as are needed for a good fit. Here we can see that the Expectation Maximisation model splits some components
arbitrarily because it is trying to fit too many components while the Dirichlet Process model adapts its number of state
automatically.
This example doesn't show it as we're in a low-dimensional space but another advantage of the Dirichlet process
model is that it can fit full covariance matrices effectively even when there are less examples per cluster than there are
dimensions in the data due to regularization properties of the inference algorithm.
```

```
scikitlearn user guide Release 0213
import itertools
import numpy as np
from scipy import linalg
import matplotlib.pyplot as plt
import matplotlib as mpl
from sklearn import mixture
coloriter = itertools.cycle([c for c in ['cornflowerblue', 'gold', 'darkorange']])

def plot_results(X, Y, means, covariances, index, title):
    plot = plt.subplot(2, 1, 1)
    index = 0
    for i, (mean, covar, color) in enumerate(zip(means, covariances, coloriter)):
        v, w = linalg.eigh(covar)
        v = 2 * np.sqrt(v)
        u = w[0]
        linalg.norm(w[0])

        # as the DP will not use every component it has access to
        # unless it needs it we shouldnt plot the redundant
        # components
        if not np.any(Y == i):
            continue

        plt.scatter(X[Y == i, 0:2], XY = i, 8, color=color)

520 Gaussian Mixture Models 1247
```

scikitlearn user guide Release 0213

Plot an ellipse to show the Gaussian component

angle nparctanu1 u0

angle 180 angle nppl convert to degrees

ell mplpatchesEllipse mean v0 v1 180 angle colorcolor

ellsetclipboxplotbbox

ellsetalpha05

plotaddartistell

pltxlim9 5

pltylim3 6

pltxticks

pltyticks

plttitletitle

Number of samples per component

nsamples 500

Generate random sample two components

nprandomseed0

C nparray0 01 17 4

X nprnpdotnprandomrandnnsamples 2 C

7nprandomrandnnsamples 2 nparray6 3

Fit a Gaussian mixture with EM using five components

gmm mixtureGaussianMixture ncomponents5 covariancetypefullfitX

plotresultsX gmmpredictX gmmmeans gmmcovariances 0

Gaussian Mixture

Fit a Dirichlet process Gaussian mixture using five components

dpgmm mixtureBayesianGaussianMixture ncomponents5

covariancetypefullfitX

plotresultsX dpgmmpredictX dpgmmmeans dpgmmcovariances 1

Bayesian Gaussian Mixture with a Dirichlet process prior

pltshow

Total running time of the script 0 minutes 0138 seconds

Note Click here to download the full example code

5203 Gaussian Mixture Model Selection

This example shows that model selection can be performed with Gaussian Mixture Models using informationtheoretic

criteria BIC Model selection concerns both the covariance type and the number of components in the model In that

case AIC also provides the right result not shown to save time but BIC is better suited if the problem is to identify

the right model Unlike Bayesian procedures such inferences are priorfree

In that case the model with 2 components and full covariance which corresponds to the true generative model is

selected

1248 Chapter 5 Examples

```
scikitlearn user guide Release 0213
import numpy as np
import itertools
from scipy import linalg
import matplotlib.pyplot as plt
import matplotlib as mpl
from sklearn import mixture
printdoc
    Number of samples per component
nsamples 500
    Generate random sample two components
nrandomseed0
C nparray0 01 17 4
X npnpdotnprandomrandnnsamples 2 C
7nprandomrandnnsamples 2 nparray6 3
lowestbic npinfy
bic
ncomponentsrange range1 7
cvtypes spherical tied diag full
forcvtype incvtypes
forncomponents inncomponentsrange
520 Gaussian Mixture Models 1249
```

```
scikitlearn user guide Release 0213
Fit a Gaussian mixture with EM
gmm mixtureGaussianMixture(ncomponents, ncomponents
covariance_type=covtype
gmmfitX
bicappendgmmbicX
ifbic1 lowestbic
lowestbic bic1
bestgmm gmm
bic nparraybic
coloriter iter_tools.cycle_color('navy', 'turquoise', 'cornflowerblue',
'darkorange')
clf bestgmm
bars
Plot the BIC scores
plt.figure(figsize=(8, 6))
spl, plt.subplot(2, 1, 1)
for i, cvtype, color in enumerate(zip(cvtypes, coloriter,
xpos, np.array(ncomponentsrange[2:]), 2)
bars.append(plt.bar(xpos, bic[i], len(ncomponentsrange[1:]),
width=2, color=color)
plt.xticks(ncomponentsrange)
plt.ylim(bicmin, 101 - bicmax, bicmax, bicmax)
plt.title('BIC score per model')
xpos, np.mod(bicargmin, len(ncomponentsrange)) - 65
2 * np.floor(bicargmin / len(ncomponentsrange))
plt.text(xpos, bicmin, '0.97 - 0.3', bicmax, fontsize=14)
spl.set_xlabel('Number of components')
spl.legend(b0, forbin, bars, cvtypes)
Plot the winner
splot, plt.subplot(2, 1, 2)
Y = clf.predict(X)
for i, mean, cov, color in enumerate(zip(clf.means_, clf.covariances_,
coloriter)):
v, w = linalg.eigh(cov)
if not np.any(Y[i] - mean):
continue
plt.scatter(XY[i], XY[i], 8, color, color)
Plot an ellipse to show the Gaussian component
angle = np.arctan2(w[0][1], w[0][0])
angle = 180 * angle / np.pi # convert to degrees
v = 2 * np.sqrt(2) * np.sqrt(v)
ell = mpatches.Ellipse(mean, v[0], v[1], 180 * angle, color, color)
ell.set_clipbox(splot.bbox)
ell.set_alpha(0.5)
splot.add_artist(ell)
plt.xticks
plt.yticks
plt.title('Selected GMM full model 2 components')
plt.subplots_adjust(hspace=0.35, bottom=0.02)
plt.show
Total running time of the script: 0 minutes 02.07 seconds
1250 Chapter 5 Examples
```


scikitlearn user guide Release 0213

Note [Click here to download the full example code](#)

5204 GMM covariances

Demonstration of several covariances types for Gaussian mixture models

SeeGaussian mixture models for more information on the estimator

Although GMM are often used for clustering we can compare the obtained clusters with the actual classes from the dataset We initialize the means of the Gaussians with the means of the classes from the training set to make this comparison valid

We plot predicted labels on both training and held out test data using a variety of GMM covariance types on the iris dataset We compare GMMs with spherical diagonal full and tied covariance matrices in increasing order of performance Although one would expect full covariance to perform best in general it is prone to overfitting on small datasets and does not generalize well to held out test data

On the plots train data is shown as dots while test data is shown as crosses The iris dataset is fourdimensional Only the first two dimensions are shown here and thus some points are separated in other dimensions

520 Gaussian Mixture Models 1251

```
scikitlearn user guide Release 0213
Author Ron Weiss ronweissgmailcom Gael Varoquaux
Modified by Thierry Guillemot thierryguillemotworkgmailcom
License BSD (Binary Search Decision) 3 clause
import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets
from sklearnmixture import GaussianMixture
from sklearnmodelselection import StratifiedKFold
printdoc
colors navy turquoise darkorange
1252 Chapter 5 Examples
```

scikitlearn user guide Release 0213

defmakeellipsesgmm ax
forn color inenumeratecolors
ifgmmcovariancetype full
covariances gmmcovariancesn2 2
elifgmmcovariancetype tied
covariances gmmcovariances2 2
elifgmmcovariancetype diag
covariances npdiaggmmcovariancesn2
elifgmmcovariancetype spherical
covariances npeyegmmmeansshape1 gmmcovariancesn
v w nplinalgeighcovariances
u w0 nplinalgnormw0
angle nparctan2u1 u0
angle 180 angle nppi convert to degrees
v 2npsqrt2 npsqrtv
ell mplpatchesEllipsegmmmeansn 2 v0 v1
180 angle colorcolor
ellsetclipboxaxbbox
ellsetalpha05
axaddartistell
axsetaspectequal datalim
iris datasetsloadiris
Break up the dataset into nonoverlapping training 75 and testing
25 sets
skf StratifiedKFoldnsplits4
Only take the first fold
trainindex testindex nextiterskfsplitirisdata iristarget
Xtrain irisdatatrainindex
ytrain iristargettrainindex
Xtest irisdatatestindex
ytest iristargettestindex
nclasses lennpuniqueytrain
Try GMMs using different types of covariances
estimators covtype GaussianMixturecomponentsnclasses
covariancetypecovtype maxiter20 randomstate0
forcovtype inspherical diag tied full
nestimators lenestimators
pltfigurefigsize3 nestimators 2 6
pltsubplotsadjustbottom01 top095 hspace15 wspace05
left01 right99
forindex name estimator inenumerateestimatorsitems
Since we have class labels for the training data we can
initialize the GMM parameters in a supervised manner
estimatormeansinit nparrayXtrainytrain imeanaxis0
foriinrangenclasses
Train the other parameters using the EM algorithm
estimatorfitXtrain
520 Gaussian Mixture Models 1253

```
scikitlearn user guide Release 0213
h pltsubplot2 nestimators 2 index 1
makeellipseestimator h
for color in enumerate(colors)
data irisdatairistarget n
pltscatterdata 0 data 1 s08 colorcolor
labeliristargetnamesn
Plot the test data with crosses
for color in enumerate(colors)
data Xtestytest n
pltscatterdata 0 data 1 markerx colorcolor
ytrainpred estimatorpredictXtrain
trainaccuracy npmean(ytrainpred.ravel() - ytrain).ravel() 100
plttext005 09 Train accuracy 1f trainaccuracy
transformhtransAxes
ytestpred estimatorpredictXtest
testaccuracy npmean(ytestpred.ravel() - ytest).ravel() 100
plttext005 08 Test accuracy 1f testaccuracy
transformhtransAxes
pltxticks
pltxticks
plttitle
pltlegendscatterpoints1 loclower right propdictsize12
pltshow
```

Total running time of the script 0 minutes 0097 seconds

Note [Click here to download the full example code](#)

5205 Gaussian Mixture Model Sine Curve

This example demonstrates the behavior of Gaussian mixture models fit on data that was not sampled from a mixture of Gaussian random variables. The dataset is formed by 100 points loosely spaced following a noisy sine curve. There is therefore no ground truth value for the number of Gaussian components.

The first model is a classical Gaussian Mixture Model with 10 components fit with the ExpectationMaximization algorithm.

The second model is a Bayesian Gaussian Mixture Model with a Dirichlet process prior fit with variational inference.

The low value of the concentration prior makes the model favor a lower number of active components. This model “decides” to focus its modeling power on the big picture of the structure of the dataset: groups of points with alternating directions modeled by nondiagonal covariance matrices. Those alternating directions roughly capture the alternating nature of the original sine signal.

The third model is also a Bayesian Gaussian mixture model with a Dirichlet process prior, but this time the value of the concentration prior is higher, giving the model more liberty to model the fine-grained structure of the data. The result is a mixture with a larger number of active components that is similar to the first model where we arbitrarily decided to fix the number of components to 10.

scikitlearn user guide Release 0213

Which model is the best is a matter of subjective judgement do we want to favor models that only capture the big picture to summarize and explain most of the structure of the data while ignoring the details or do we prefer models that closely follow the high density regions of the signal

The last two panels show how we can sample from the last two models The resulting samples distributions do not look exactly like the original data distribution The difference primarily stems from the approximation error we made by using a model that assumes that the data was generated by a finite number of Gaussian components instead of a continuous noisy sine curve

```
import itertools
import numpy as np
from scipy import linalg
import matplotlib.pyplot as plt
import matplotlib as mpl
520 Gaussian Mixture Models 1255
```

```
scikitlearn user guide Release 0213
from sklearn import mixture
printdoc
coloriter itertoolscyclenavy c cornflowerblue gold
darkorange
defplotresultsX Y means covariances index title
splot pltsubplot5 1 1 index
fori mean covar color inenumeratezip
means covariances coloriter
v w linalgeighcovar
v 2npsqrt2 npsqrtv
u w0 linalgnormw0
as the DP will not use every component it has access to
unless it needs it we shouldnt plot the redundant
components
if notnpanyY i
continue
pltscatterXY i 0 XY i 1 8 colorcolor
Plot an ellipse to show the Gaussian component
angle nparctanu1 u0
angle 180 angle nppl convert to degrees
ell mplpatchesEllipsemean v0 v1 180 angle colorcolor
ellsetclipboxsplotbbox
ellsetalpha05
splotaddartistell
pltxlim6 4 nppl 6
pltylim5 5
plttitletitle
pltxticks
pltxticks
defplotsamplesX Y ncomponents index title
pltsubplot5 1 4 index
fori color inziprangenccomponents coloriter
as the DP will not use every component it has access to
unless it needs it we shouldnt plot the redundant
components
if notnpanyY i
continue
pltscatterXY i 0 XY i 1 8 colorcolor
pltxlim6 4 nppl 6
pltylim5 5
plttitletitle
pltxticks
pltxticks
Parameters
nsamples 100
1256 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
Generate random sample following a sine curve
nprandomseed0
X npzerosnsamples 2
step 4 nppi nsamples
foriinrangeXshape0
x istep 6
Xi 0 x nprandomnormal0 01
Xi 1 3 npsinx nprandomnormal0 2
pltfigurefigsize10 10
pltsubplotsadjustbottom04 top095 hspace2 wspace05
left03 right97
Fit a Gaussian mixture with EM using ten components
gmm mixtureGaussianMixturencomponents10 covariancetypefull
maxiter100fitX
plotresultsX gmmpredictX gmmmeans gmmcovariances 0
Expectationmaximization
dpgmm mixtureBayesianGaussianMixture
ncomponents10 covariancetypefull weightconcentrationprior1e2
weightconcentrationpriortypedirichletprocess
meanprecisionprior1e2 covarianceprior1e0 npeye2
initparamsrandom maxiter100 randomstate2fitX
plotresultsX dpgmmpredictX dpgmmmeans dpgmmcovariances 1
Bayesian Gaussian mixture models with a Dirichlet process prior
rfor gamma0001
Xs ys dpgmmsamplensamples2000
plotsamplesXs ys dpgmmncomponents 0
Gaussian mixture with a Dirichlet process prior
rfor gamma0001 sampled with 2000 samples
dpgmm mixtureBayesianGaussianMixture
ncomponents10 covariancetypefull weightconcentrationprior1e2
weightconcentrationpriortypedirichletprocess
meanprecisionprior1e2 covarianceprior1e0 npeye2
initparamskmeans maxiter100 randomstate2fitX
plotresultsX dpgmmpredictX dpgmmmeans dpgmmcovariances 2
Bayesian Gaussian mixture models with a Dirichlet process prior
rfor gamma0100
Xs ys dpgmmsamplensamples2000
plotsamplesXs ys dpgmmncomponents 1
Gaussian mixture with a Dirichlet process prior
rfor gamma0100 sampled with 2000 samples
pltshow
Total running time of the script 0 minutes 0301 seconds
Note Click here to download the full example code
520 Gaussian Mixture Models 1257
```

5206 Concentration Prior Type Analysis of Variation Bayesian Gaussian Mixture

This example plots the ellipsoids obtained from a toy dataset mixture of three Gaussians fit

ted by the BayesianGaussianMixture class models with a Dirichlet distribution prior

weightconcentrationpriortypedirichletdistribution and a Dirichlet process

prior weightconcentrationpriortypedirichletprocess On each figure we plot the

results for three different values of the weight concentration prior

TheBayesianGaussianMixture class can adapt its number of mixture components automatically The param

eterweightconcentrationprior has a direct link with the resulting number of components with nonzero

weights Specifying a low value for the concentration prior will make the model put most of the weight on few com

ponents set the remaining components weights very close to zero High values of the concentration prior will allow a

larger number of components to be active in the mixture

The Dirichlet process prior allows to define an infinite number of components and automatically selects the correct

number of components it activates a component only if it is necessary

On the contrary the classical finite mixture model with a Dirichlet distribution prior will favor more uniformly weighted

components and therefore tends to divide natural clusters into unnecessary subcomponents

•

scikitlearn user guide Release 0213

```
•
Author Thierry Guillemot thierryguillemotworkgmailcom
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
from sklearnmixture import BayesianGaussianMixture
printdoc
defplotellipsesax weights means covars
forninrangemeansshape0
eigvals eigvecs nplinalgeighcovarsn
uniteigvec eigvecs0 nplinalgnormeigvecs0
angle nparctan2uniteigvec1 uniteigvec0
Ellipse needs degrees
angle 180 angle nppi
eigenvector normalization
eigvals 2 npsqrt2 npsqrteigvals
ell mplpatchesEllipsemeansn eigvals0 eigvals1
180 angle edgcolorblack
ellsetclipboxaxbbox
ellsetalphaweightsn
ellsetfacecolor56B4E9
axaddartistell
defplotresultsax1 ax2 estimator X y title plottitleFalse
ax1settitletitle
ax1scatterX 0 X 1 s5 markero colorcolorsy alpha08
ax1setxlim2 2
ax1setylim3 3
520 Gaussian Mixture Models 1259
```

scikitlearn user guide Release 0213
ax1setxticks
ax1setyticks
plotellipsesax1 estimatorweights estimatormeans
estimatorcovariances
ax2getxaxissettickparamsdirectionout
ax2yaxisgridTrue alpha07
fork winenumerateestimatorweights
ax2bark w width09 color56B4E9 zorder3
aligncenter edgecolorblack
ax2textk w 0007 1f w100
horizontalalignmentcenter
ax2setxlim6 2 ncomponents 4
ax2setylim0 11
ax2tickparamsaxisy whichboth leftFalse
rightFalse labelleftFalse
ax2tickparamsaxisx whichboth topFalse
ifplottitle
ax1setylabelEstimated Mixtures
ax2setylabelWeight of each component
Parameters of the dataset
randomstate ncomponents nfeatures 2 3 2
colors nparray0072B2 F0E442 D55E00
covars nparray7 0 0 1
5 0 0 1
5 0 0 1
samples nparray200 500 200
means nparray0 70
0 0
0 70
meanprecisionprior 08 to minimize the influence of the prior
estimators
Finite mixture with a Dirichlet distribution nprior and
rgamma0 BayesianGaussianMixture
weightconcentrationpriortypedirichletdistribution
ncomponents2 ncomponents regcovar0 initparamsrandom
maxiter1500 meanprecisionprior8
randomstaterandomstate 0001 1 1000
Infinite mixture with a Dirichlet process nprior and rgamma0
BayesianGaussianMixture
weightconcentrationpriortypedirichletprocess
ncomponents2 ncomponents regcovar0 initparamsrandom
maxiter1500 meanprecisionprior8
randomstaterandomstate 1 1000 100000
Generate data
rng nprandomRandomStaterandomstate
X npvstack
rngmultivariatenormalmeansj covarsj samplesj
forjinrangencomponents
y npconcatenatenpfullsamplesj j dtypeint
forjinrangencomponents
Plot results in two different figures
1260 Chapter 5 Examples

scikitlearn user guide Release 0213
fortitle estimator concentrationsprior inestimators
pltfigurefigsize47 3 8
pltsubplotsadjustbottom04 top090 hspace05 wspace05
left03 right99
gs gridspecGridSpec3 lenconcentrationsprior
fork concentration inenumerateconcentrationsprior
estimatorweightconcentrationprior concentration
estimatorfitX
plotresultspltsubplotgs02 k pltsubplots2 k estimator
X y r s1e title concentration
plttitlek 0
pltshow
Total running time of the script 0 minutes 6923 seconds
521 Model Selection
Examples related to the sklearnmodelselection module
Note Click here to download the full example code
5211 Plotting CrossValidated Predictions
This example shows how to use crossvalpredict to visualize prediction errors
521 Model Selection 1261

```
scikitlearn user guide Release 0213
from sklearn import datasets
from sklearn.model_selection import cross_val_predict
from sklearn import linear_model
import matplotlib.pyplot as plt
lr = linear_model.LinearRegression()
boston = datasets.load_boston()
y = boston.target
cross_val_predict returns an array of the same size as y where each entry
is a prediction obtained by cross validation
predicted = cross_val_predict(lr, boston.data, y, cv=10)
fig, ax = plt.subplots()
ax.scatter(y, predicted, edgecolors=(0, 0, 0))
ax.plot(ymin, ymax, ymin, ymax, k='lw4')
ax.set_xlabel('Measured')
ax.set_ylabel('Predicted')
plt.show
```

Total running time of the script: 0 minutes 00.24 seconds
Note: Click [here](#) to download the full example code
1262 Chapter 5 Examples

scikitlearn user guide Release 0213

5212 Plotting Validation Curves

In this plot you can see the training scores and validation scores of an SVM (Support Vector Machine) for different values of the kernel parameter gamma For very low values of gamma you can see that both the training score and the validation score are low This is called underfitting Medium values of gamma will result in high values for both scores ie the classifier is performing fairly well If gamma is too high the classifier will overfit which means that the training score is good but the validation score is poor

```
printdoc
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_digits
from sklearn.svm import SVC (Support Vector Classification)
from sklearn.model_selection import validation_curve
digits = load_digits
X, y = digits.data, digits.target
param_range = np.logspace(0, 1, 5)
train_scores, test_scores = validation_curve
SVC (Support Vector Classification) X, y, param_name=gamma, param_range=param_range,
cv=5, scoring=accuracy, n_jobs=1
train_scores_mean = np.mean(train_scores, axis=1)
train_scores_std = np.std(train_scores, axis=1)
521 Model Selection 1263
```

```
scikitlearn user guide Release 0213
testscoresmean npmeantestscores axis1
testscoresstd npstdtestscores axis1
plttitleValidation Curve with SVM (Support Vector Machine)
pltxlabelgamma
pltylabelScore
pltylim00 11
lw 2
pltsemilogxparamrange trainscoresmean labelTraining score
colordarkorange lwlw
pltfillbetweenparamrange trainscoresmean trainscoresstd
trainscoresmean trainscoresstd alpha02
colordarkorange lwlw
pltsemilogxparamrange testscoresmean labelCrossvalidation score
colornavy lwlw
pltfillbetweenparamrange testscoresmean testscoresstd
testscoresmean testscoresstd alpha02
colornavy lwlw
pltlegendlocbest
pltshow
```

Total running time of the script 0 minutes 13416 seconds

Note Click here to download the full example code

5213 Underfitting vs Overfitting

This example demonstrates the problems of underfitting and overfitting and how we can use linear regression with polynomial features to approximate nonlinear functions The plot shows the function that we want to approximate which is a part of the cosine function In addition the samples from the real function and the approximations of different models are displayed The models have polynomial features of different degrees We can see that a linear function polynomial with degree 1 is not sufficient to fit the training samples This is called underfitting A polynomial of degree 4 approximates the true function almost perfectly However for higher degrees the model will overfit the training data ie it learns the noise of the training data We evaluate quantitatively overfitting underfitting by using crossvalidation We calculate the mean squared error MSE on the validation set the higher the less likely the model generalizes correctly from the training data

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import cross_val_score
def truefun(X):
    return np.cos(15 * np.pi * X)
nprandomseed0
nsamples = 30
degrees = [1, 4, 15]
X = np.sort(np.random.rand(nsamples))
y = truefun(X)
nprandomrandnsamples01
plt.figure(figsize=(14, 5))
for i in range(len(degrees)):
    ax = plt.subplot(1, len(degrees), i + 1)
    plt.setp(ax, xticks=degrees[i], yticks=[])
    polynomial_features = PolynomialFeatures(degree=degrees[i],
                                             include_bias=False)
    linear_regression = LinearRegression()
    pipeline = Pipeline([
        ('polynomial_features', polynomial_features),
        ('linear_regression', linear_regression)
    ])
    pipeline.fit(X, y)
    # Evaluate the models using cross-validation
    scores = cross_val_score(pipeline, X, y, scoring='neg_mean_squared_error',
                              cv=10)
    X_test = np.linspace(0, 1, 100)
    plt.plot(X_test, pipeline.predict(X_test), np.newaxis, label='Model')
    plt.plot(X_test, truefun(X_test), label='True function')
    plt.scatter(X, y, edgecolor='b', s=20, label='Samples')
    plt.xlabel(x)
    plt.ylabel(y)
    plt.xlim(0, 1)
    plt.ylim(2, 2)
    plt.legend(loc='best')
    plt.title('Degree: %d, nMSE: %2e' % (degrees[i], scores.mean()))
    plt.show()
degrees[i], scores.mean(), scores.std()
plt.show()
Total running time of the script: 0 minutes 00.84 seconds
Note: Click here to download the full example code
521 Model Selection 1265
```

scikitlearn user guide Release 0213

5214 Parameter estimation using grid search with crossvalidation

This examples shows how a classifier is optimized by crossvalidation which is done using the sklearn
modelselectionGridSearchCV object on a development set that comprises only half of the available labeled
data

The performance of the selected hyperparameters and trained model is then measured on a dedicated evaluation set
that was not used during the model selection step

More details on tools available for model selection can be found in the sections on Crossvalidation evaluating
estimator performance andTuning the hyperparameters of an estimator

Out

Tuning hyperparameters for precision

Best parameters set found on development set

C 10 gamma 0001 kernel rbf

Grid scores on development set

0986 0016 for C 1 gamma 0001 kernel rbf

0959 0029 for C 1 gamma 00001 kernel rbf

0988 0017 for C 10 gamma 0001 kernel rbf

0982 0026 for C 10 gamma 00001 kernel rbf

0988 0017 for C 100 gamma 0001 kernel rbf

0982 0025 for C 100 gamma 00001 kernel rbf

0988 0017 for C 1000 gamma 0001 kernel rbf

0982 0025 for C 1000 gamma 00001 kernel rbf

0975 0014 for C 1 kernel linear

0975 0014 for C 10 kernel linear

0975 0014 for C 100 kernel linear

0975 0014 for C 1000 kernel linear

Detailed classification report

The model is trained on the full development set

The scores are computed on the full evaluation set

precision recall f1score support

0 100 100 100 89

1 097 100 098 90

2 099 098 098 92

3 100 099 099 93

4 100 100 100 76

5 099 098 099 108

6 099 100 099 89

7 099 100 099 78

8 100 098 099 92

9 099 099 099 92

accuracy 099 899

macro avg 099 099 099 899

weighted avg 099 099 099 899

Tuning hyperparameters for recall

1266 Chapter 5 Examples

scikitlearn user guide Release 0213
Best parameters set found on development set
C 10 gamma 0001 kernel rbf
Grid scores on development set
0986 0019 for C 1 gamma 0001 kernel rbf
0957 0029 for C 1 gamma 00001 kernel rbf
0987 0019 for C 10 gamma 0001 kernel rbf
0981 0028 for C 10 gamma 00001 kernel rbf
0987 0019 for C 100 gamma 0001 kernel rbf
0981 0026 for C 100 gamma 00001 kernel rbf
0987 0019 for C 1000 gamma 0001 kernel rbf
0981 0026 for C 1000 gamma 00001 kernel rbf
0972 0012 for C 1 kernel linear
0972 0012 for C 10 kernel linear
0972 0012 for C 100 kernel linear
0972 0012 for C 1000 kernel linear
Detailed classification report
The model is trained on the full development set
The scores are computed on the full evaluation set
precision recall f1score support
0 100 100 100 89
1 097 100 098 90
2 099 098 098 92
3 100 099 099 93
4 100 100 100 76
5 099 098 099 108
6 099 100 099 89
7 099 100 099 78
8 100 098 099 92
9 099 099 099 92
accuracy 099 899
macro avg 099 099 099 899
weighted avg 099 099 099 899
from sklearn import datasets
from sklearnmodelselection import traintestsplit
from sklearnmodelselection import GridSearchCV
from sklearnmetrics import classificationreport
from sklearnsvm import SVC (Support Vector Classification)
printdoc
Loading the Digits dataset
digits datasetsloadaddigits
521 Model Selection 1267

```
scikitlearn user guide Release 0213
To apply an classifier on this data we need to flatten the image to
turn the data in a samples feature matrix
nsamples = len(digits.images)
X = digits.images.reshape((nsamples, 1
y = digits.target

Split the dataset in two equal parts
Xtrain, Xtest, ytrain, ytest = train_test_split
X, y, test_size=0.5, random_state=0

Set the parameters by crossvalidation
tuned_parameters = {'kernel': 'rbf', 'gamma': 1e-3, '1e4
C = 1, 10, 100, 1000
kernel = 'linear', C = 1, 10, 100, 1000

scores = precision_recall
for score in scores:
    print('Tuning hyperparameters for %s: %s' % (score,
    print('clf: %s' % GridSearchCV(SVC(), tuned_parameters, cv=5,
    scoring = 'smacro', score
    clf.fit(Xtrain, ytrain)
    print('Best parameters set found on development set: %s' %
    print('print: %s' % clf.best_params_)
    print('Grid scores on development set: %s' %
    print('means: %s' % clf.cv_results_.mean_test_score)
    print('stds: %s' % clf.cv_results_.std_test_score)
    print('mean std: %s' % zip(means, stds, clf.cv_results_.params))
    print('0.3f: %s' % forr)
    print('mean std: 2 params')
    print('Detailed classification report: %s' %
    print('The model is trained on the full development set: %s' %
    print('The scores are computed on the full evaluation set: %s' %
    print('ytrue: %s, ypred: %s, ytest: %s, clf.predict(Xtest): %s' %
    print('classification report: %s' % ytrue, ypred)
    print('Note: the problem is too easy, the hyperparameter plateau is too flat and the
    output model is the same for precision and recall with ties in quality
    Total running time of the script: 0 minutes 43.44 seconds
    Note: Click here to download the full example code
    1268 Chapter 5 Examples
```

scikitlearn user guide Release 0213

5215 Train error vs Test error

Illustration of how the performance of an estimator on unseen data test data is not the same as the performance on training data As the regularization increases the performance on train decreases while the performance on test is optimal within a range of values of the regularization parameter The example with an ElasticNet regression model and the performance is measured using the explained variance aka R2

Out

Optimal regularization parameter 000013141473626117567

printdoc

Author Alexandre Gramfort alexandregamfortinriafr

License BSD (Binary Search Decision) 3 clause

import numpy as np

from sklearn import linearmodel

Generate sample data

521 Model Selection 1269

```
scikitlearn user guide Release 0213
nsamplestrain nsamplestest nfeatures 75 150 500
nprandomseed0
coef nprandomrandnnfeatures
coef50 00 only the top 10 features are impacting the model
X nprandomrandnnnsamplestrain nsamplestest nfeatures
y npdotX coef
Split train and test data
Xtrain Xtest Xnsamplestrain Xnsamplestrain
ytrain ytest ynsamplestrain ynsamplestrain

Compute train and test errors
alphas nplogspace5 1 60
enet linearmodelElasticNetl1ratio07 maxiter10000
trainerrors list
testerrors list
foralphainalphas
enetsetparamsalphaalpha
enetfitXtrain ytrain
trainerrorsappendenetscoreXtrain ytrain
testerrorsappendenetscoreXtest ytest
ialphaoptim npargmaxtesterrors
alphaoptim alphasialphaoptim
printOptimal regularization parameter s alphaoptim
Estimate the coef on full data with optimal regularization parameter
enetsetparamsalphaalphaoptim
coef enetfitX ycoef

Plot results functions
import matplotlib.pyplot as plt
pltsubplot2 1 1
pltsemilogxalphas trainerrors labelTrain
pltsemilogxalphas testerrors labelTest
pltvlinesalphaoptim pltylim0 npmaxtesterrors colork
linewidth3 labelOptimum on test
pltlegendloclower left
pltylim0 12
pltxlabelRegularization parameter
pltylabelPerformance
Show estimated coef vs true coef
pltsubplot2 1 2
pltplotcoef labelTrue coef
pltplotcoef labelEstimated coef
pltlegend
pltsubplotsadjust009 004 094 094 026 026
pltshow
Total running time of the script 0 minutes 3507 seconds
Note Click here to download the full example code
1270 Chapter 5 Examples
```

scikitlearn user guide Release 0213

5216 Receiver Operating Characteristic ROC with cross validation

Example of Receiver Operating Characteristic ROC metric to evaluate classifier output quality using crossvalidation
ROC curves typically feature true positive rate on the Y axis and false positive rate on the X axis This means that the top left corner of the plot is the “ideal” point a false positive rate of zero and a true positive rate of one This is not very realistic but it does mean that a larger area under the curve AUC is usually better

The “steepness” of ROC curves is also important since it is ideal to maximize the true positive rate while minimizing the false positive rate

This example shows the ROC response of different datasets created from Kfold crossvalidation Taking all of these curves it is possible to calculate the mean area under curve and see the variance of the curve when the training set is split into different subsets This roughly shows how the classifier output is affected by changes in the training data and how different the splits generated by Kfold crossvalidation are from one another

Note
See also `sklearnmetricsrocaucscore` `sklearnmodelselectioncrossvalscore`

Receiver Operating Characteristic ROC

```
printdoc
import numpy as np
from scipy import interp
521 Model Selection 1271
```

```
scikitlearn user guide Release 0213
import matplotlib.pyplot as plt
from sklearn import svm datasets
from sklearnmetrics import roccurve auc
from sklearnmodelselection import StratifiedKFold

Data IO and generation
Import some data to play with
iris datasetsloadiris
X irisdata
y iristarget
X y Xy 2 yy 2
nsamples nfeatures Xshape
Add noisy features
randomstate np.random.RandomState(0)
X np.concatenate((X, random.randn(nsamples, 200 - nfeatures)))

Classification and ROC analysis
Run classifier with crossvalidation and plot ROC curves
cv = StratifiedKFold(n_splits=6)
classifier = svm.SVC(kernel='linear', probability=True)
random_state = randomstate
tprs, aucs, meanfpr = np.linspace(0, 1, 100)
for i in range(0, 100):
    for train, test in cv.split(X, y):
        probas = classifier.fit(X[train], y[train]).predict_proba(X[test])
        # Compute ROC curve and area the curve
        fpr, tpr, thresholds = roc_curve(y_test, probas[:, 1])
        tprs.append(interp(meanfpr, fpr, tpr))
        tprs[-1] = 0.0
        rocauc = auc(fpr, tpr)
        aucs.append(rocauc)
    plt.plot(fpr, tpr, lw=1, alpha=0.3)
    label = 'ROC fold %02f' % (i + 1)
    plt.plot([0, 1], [0, 1], linestyle='--', lw=2, color='r',
             label='Chance', alpha=0.8)
    mean_tpr = np.mean(tprs, axis=0)
    mean_tpr[-1] = 1.0
    mean_auc = auc(meanfpr, mean_tpr)
    std_auc = np.std(aucs)
    plt.plot(meanfpr, mean_tpr, color='b',
             label='Mean ROC AUC %02fpm02f' % (mean_auc, std_auc),
             lw=2, alpha=0.8)
    std_tpr = np.std(tprs, axis=0)
    tprs_upper = np.minimum(mean_tpr + std_tpr, 1)
    tprs_lower = np.maximum(mean_tpr - std_tpr, 0)
    plt.fill_between(meanfpr, tprs_lower, tprs_upper, color='grey', alpha=0.2)
plt.show()
```

scikitlearn user guide Release 0213
tprslower npmaximummeantpr stdtpr 0
pltfillbetweenmeanfpr tprslower tprsupper colorgrey alpha2
labelrpm 1 std dev
pltxlim005 105
pltylim005 105
pltxlabelFalse Positive Rate
pltylabelTrue Positive Rate
plttitleReceiver operating characteristic example
pltlegendloclower right
pltshow
Total running time of the script 0 minutes 0245 seconds
Note Click here to download the full example code
5217 Comparing randomized search and grid search for hyperparameter estimation
Compare randomized search and grid search for optimizing hyperparameters of a random forest All parameters that influence the learning are searched simultaneously except for the number of estimators which poses a time quality tradeoff
The randomized search and the grid search explore exactly the same space of parameters The result in parameter settings is quite similar while the run time for randomized search is drastically lower
The performance is slightly worse for the randomized search though this is most likely a noise effect and would not carry over to a heldout test set
Note that in practice one would not search over this many different parameters simultaneously using grid search but pick only the ones deemed most important
Out
RandomizedSearchCV took 442 seconds for 20 candidates parameter settings
Model with rank 1
Mean validation score 0939 std 0024
Parameters bootstrap False criterion entropy maxdepth None max
↔features 7 minsamplesplit 3
Model with rank 2
Mean validation score 0933 std 0022
Parameters bootstrap False criterion gini maxdepth None maxfeatures
↔6 minsamplesplit 6
Model with rank 3
Mean validation score 0930 std 0031
Parameters bootstrap True criterion gini maxdepth None maxfeatures
↔6 minsamplesplit 6
GridSearchCV took 1324 seconds for 72 candidate parameter settings
Model with rank 1
Mean validation score 0937 std 0019
Parameters bootstrap False criterion entropy maxdepth None max
↔features 10 minsamplesplit 2
521 Model Selection 1273

```
scikitlearn user guide Release 0213
Model with rank 2
Mean validation score 0936 std 0020
Parameters bootstrap False criterion gini maxdepth None maxfeatures
↔ 10 minsamplesplit 2
Model with rank 3
Mean validation score 0931 std 0029
Parameters bootstrap False criterion entropy maxdepth None max
↔ features 10 minsamplesplit 3
printdoc
import numpy as np
from time import time
from scipystats import randint assprandint
from sklearnmodelselection import GridSearchCV
from sklearnmodelselection import RandomizedSearchCV
from sklearndatasets import loaddigits
from sklearnensemble import RandomForestClassifier
    get some data
digits loaddigits
X y digitsdata digitstarget
    build a classifier
clf RandomForestClassifiernestimators20
    Utility function to report best scores
defreportresults ntop3
foriinrange1 ntop 1
candidates npflatnonzeroreultsrnktestscore i
forcandidate incandidates
printModel with rank 0formati
printMean validation score 03f std 13fformat
resultsmeantestscorecandidate
resultssdttestscorecandidate
printParameters 0formatresultsparamscandidate
print
    specify parameters and distributions to sample from
paramdist maxdepth 3 None
maxfeatures sprandint1 11
minsamplesplit sprandint2 11
bootstrap True False
criterion gini entropy
    run randomized search
nitersearch 20
randomsearch RandomizedSearchCVclf paramdistributionsparamdist
1274 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
nitersearch cv5 iidFalse
start time
randomsearchfitX y
printRandomizedSearchCV took 2fseconds for dcandidates
parameter settings time start nitersearch
reportrandomsearchcvresults
use a full grid over all parameters
paramgrid maxdepth 3 None
maxfeatures 1 3 10
minsamplesplit 2 3 10
bootstrap True False
criterion gini entropy
run grid search
gridsearch GridSearchCVclf paramgridparamgrid cv5 iidFalse
start time
gridsearchfitX y
printGridSearchCV took 2fseconds for dcandidate parameter settings
time start lengridsearchcvresultsparms
reportgridsearchcvresults
Total running time of the script 0 minutes 17712 seconds
Note Click here to download the full example code
5218 Nested versus nonnested crossvalidation
This example compares nonnested and nested crossvalidation strategies on a classifier of the iris data set Nested
crossvalidation CV (Cross-Validation) is often used to train a model in which hyperparameters also need to be optimized Nested CV (Cross-Validation)
estimates the generalization error of the underlying model and its hyperparameter search Choosing the parameters
that maximize nonnested CV (Cross-Validation) biases the model to the dataset yielding an overlyoptimistic score
Model selection without nested CV (Cross-Validation) uses the same data to tune model parameters and evaluate model performance
Information may thus “leak” into the model and overfit the data The magnitude of this effect is primarily dependent
on the size of the dataset and the stability of the model See Cawley and Talbot1for an analysis of these issues
To avoid this problem nested CV (Cross-Validation) effectively uses a series of trainvalidationtest set splits In the inner loop
here executed by GridSearchCV the score is approximately maximized by fitting a model to each training
set and then directly maximized in selecting hyperparameters over the validation set In the outer loop here in
crossvalscore generalization error is estimated by averaging test set scores over several dataset splits
The example below uses a support vector classifier with a nonlinear kernel to build a model with optimized hyperpa
rameters by grid search We compare the performance of nonnested and nested CV (Cross-Validation) strategies by taking the difference
between their scores
See Also
•Crossvalidation evaluating estimator performance
1Cawley GC Talbot NLC On overfitting in model selection and subsequent selection bias in performance evaluation J Mach Learn Res
201011 20792107
521 Model Selection 1275
```

```
scikitlearn user guide Release 0213
•Tuning the hyperparameters of an estimator
References
Out
Average difference of 0007581 with std dev of 0007833
from sklearndatasets import loadiris
from matplotlib import pyplotasplt
from sklearnsvm import SVC (Support Vector Classification)
from sklearnmodelselection import GridSearchCV crossvalscore KFold
import numpy as np
printdoc
  Number of random trials
NUMTRIALS 30
1276 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
Load the dataset
iris loadiris
Xiris irisdata
yiris iristarget
Set up possible values of parameters to optimize over
pgrid C 1 10 100
gamma 01 1
We will use a Support Vector Classifier with rbf kernel
svm SVCkernelrbf
Arrays to store scores
nonnestedscores npzerosNUMTRIALS
nestedscores npzerosNUMTRIALS
Loop for each trial
foriinrangeNUMTRIALS
Choose crossvalidation techniques for the inner and outer loops
independently of the dataset
Eg GroupKFold LeaveOneOut LeaveOneGroupOut etc
innercv KFoldnsplits4 shuffleTrue randomstatei
outercv KFoldnsplits4 shuffleTrue randomstatei
Nonnested parameter search and scoring
clf GridSearchCVestimatorsvm paramgridpgrid cvinnercv
iidFalse
clffitXiris yiris
nonnestedscoresi clfbestscore
Nested CV (Cross-Validation) with parameter optimization
nestedscore crossvalscoreclf XXiris yyiris cvoutercv
nestedscoresi nestedscoremean
scoredifference nonnestedscores nestedscores
printAverage difference of 6f with std dev of 6f
formatscoredifferencecmean scoredifferencecstd
Plot scores on each trial for nested and nonnested CV (Cross-Validation)
pltfigure
pltsubplot211
nonnestedscoresline pltplotnonnestedscores colorr
nestedline pltplotnestedscores colorb
pltylabelscore fontsize14
pltlegendnonnestedscoresline nestedline
NonNested CV (Cross-Validation) Nested CV (Cross-Validation)
bboxtoanchor0 4 5 0
plttitleNonNested and Nested Cross Validation on Iris Dataset
x5 y11 fontsize15
Plot bar chart of the difference
pltsubplot212
differenceplot pltbarrangeNUMTRIALS scoredifference
pltxlabelIndividual Trial
pltlegenddifferenceplot
521 Model Selection 1277
```

scikitlearn user guide Release 0213

NonNested CV (Cross-Validation) Nested CV (Cross-Validation) Score

bboxtoanchor0 1 8 0

plt.ylabel(score difference, fontsize=14)

plt.show()

Total running time of the script: 0 minutes 34.47 seconds

Note: Click [here](#) to download the full example code

5219 Demonstration of multimetric evaluation on cross-validation and Grid

SearchCV

Multiple metric parameter search can be done by setting the scoring parameter to a list of metric scorer names or a dict mapping the scorer names to the scorer callables

The scores of all the scorers are available in the cv_results dict at keys ending in scorer_name

mean_test_precision rank_test_precision etc

The best estimator, best_index, best_score and best_params correspond to the scorer key

that is set to the refit attribute

Author: Raghav RV, rvraghav93@gmail.com

License: BSD (Binary Search Decision)

import numpy as np

from matplotlib import pyplot as plt

from sklearn.datasets import make_hastie102

from sklearn.model_selection import GridSearchCV

from sklearn.metrics import make_scorer

from sklearn.metrics import accuracy_score

from sklearn.tree import DecisionTreeClassifier

print doc

Running GridSearchCV using multiple evaluation metrics

X, y = make_hastie102(n_samples=8000, random_state=42)

The scorers can be either be one of the predefined metric strings or a scorer callable like the one returned by make_scorer

scoring = AUC, roc_auc, Accuracy, make_scorer(accuracy_score)

Setting refit=AUC refits an estimator on the whole dataset with the parameter setting that has the best cross-validated AUC score

That estimator is made available at gs.best_estimator_ along with

parameters like gs.best_score_, gs.best_params_ and

gs.best_index_

gs = GridSearchCV(DecisionTreeClassifier(), random_state=42,

param_grid=dict(min_samples_split=range(2, 403, 10),

scoring=scoring, cv=5, refit=AUC, return_train_score=True)

1278 Chapter 5 Examples

```
scikitlearn user guide Release 0213
gsfitX y
results gscvresults
Plotting the result
pltfigurefigsize13 13
plttitleGridSearchCV evaluating using multiple scorers simultaneously
fontsize16
pltxlabelminsamplessplit
pltylabelScore
ax pltgca
axsetxlim0 402
axsetylim073 1
    Get the regular numpy array from the MaskedArray
Axis nparrayresultsparmminsamplessplitdata dtypefloat
forscorer color inzipsortedscoring g k
forsample style intrain test
samplescoremean resultsmean ss sample scorer
samplescorestd resultsstd ss sample scorer
axfillbetweenXaxis samplescoremean samplescorestd
samplescoremean samplescorestd
alpha01 ifsample test else0 colorcolor
axplotXaxis samplescoremean style colorcolor
alpha1 ifsample test else07
labelss scorer sample
bestindex npnonzeroreultsrnktest s scorer 100
bestscore resultsmeantest s scorerbestindex
    Plot a dotted vertical line at the best score for that scorer marked by x
axplotXaxisbestindex 2 0 bestscore
linestyle colorcolor markerx markeredgewidth3 ms8
    Annotate the best score for that scorer
axannotate 02f bestscore
Xaxisbestindex bestscore 0005
pltlegendlocbest
pltgridFalse
pltshow
521 Model Selection 1279
```

scikitlearn user guide Release 0213

Total running time of the script 0 minutes 20458 seconds

Note [Click here to download the full example code](#)

52110 Balance model complexity and crossvalidated score

This example balances model complexity and crossvalidated score by finding a decent accuracy within 1 standard deviation of the best accuracy score while minimising the number of PCA (Principal Component Analysis) components 1

The figure shows the tradeoff between crossvalidated score and the number of PCA (Principal Component Analysis) components The balanced case is when ncomponents6 and accuracy080 which falls into the range within 1 standard deviation of the best accuracy score

1280 Chapter 5 Examples

```
scikitlearn user guide Release 0213
1 Hastie T Tibshirani R,, Friedman J 2001 Model Assessment and Selection The Elements of Statistical
Learning pp 219260 New York NY USA Springer New York Inc
Out
The bestindex is 2
The ncomponents selected is 6
The corresponding accuracy score is 080
Author Wenhao Zhang wenhaozuclaedu
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_digits
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.model_selection import GridSearchCV
from sklearn.pipeline import Pipeline
from sklearn.svm import LinearSVC
521 Model Selection 1281
```

scikitlearn user guide Release 0213
deflowerboundcvresults

Calculate the lower bound within 1 standard deviation
of the best meantestscores
Parameters

cvresults dict of numpymasked ndarrays
See attribute cvresults of GridSearchCV
Returns

float
Lower bound within 1 standard deviation of the
best meantestscore

bestscoreidx npargmaxcvresultsmeantestscore
returncvresultsmeantestscorebestscoreidx
cvresultssdttestscorebestscoreidx
defbestlowcomplexitycvresults

Balance model complexity with crossvalidated score
Parameters

cvresults dict of numpymasked ndarrays
See attribute cvresults of GridSearchCV
Return

int
Index of a model that has the fewest PCA (Principal Component Analysis) components
while has its test score within 1 standard deviation of the best
meantestscore

threshold lowerboundcvresults
candidateidx npflatnonzerocvresultsmeantestscore threshold
bestidx candidateidxcvresultsparmreducedimncomponents
candidateidxargmin
returnbestidx
pipe Pipeline
reducedim PCArandomstate42
classify LinearSVCrandomstate42

paramgrid
reducedimncomponents 2 4 6 8

grid GridSearchCVpipe cv10 njobs1 paramgridparamgrid
scoringaccuracy refitbestlowcomplexity
1282 Chapter 5 Examples


```
scikitlearn user guide Release 0213
digits loaddigits
gridfitdigitsdata digitstarget
ncomponents gridcvresultsparmreducedimncomponents
testscores gridcvresultsmeanestestscore
pltfigure
pltbarncomponents testscores width13 colorb
lower lowerboundgridcvresults
pltaxhlinenpmaxtestscores linestyle colory
labelBest score
pltaxhlinelower linestyle color5 labelBest score 1 std
plttitleBalance model complexity and crossvalidated score
pltxlabelNumber of PCA (Principal Component Analysis) components used
pltylabelDigit classification accuracy
pltxticksncomponentstolist
ptylim0 10
pltlegendlocupper left
bestindex gridbestindex
printThe bestindex is d bestindex
printThe ncomponents selected is d ncomponentsbestindex
printThe corresponding accuracy score is 2f
gridcvresultsmeanestestscorebestindex
pltshow
Total running time of the script 0 minutes 16219 seconds
Note Click here to download the full example code
52111 Sample pipeline for text feature extraction and evaluation
The dataset used in this example is the 20 newsgroups dataset which will be automatically downloaded and then cached
and reused for the document classification example
You can adjust the number of categories by giving their names to the dataset loader or setting them to None to get the
20 of them
Here is a sample output of a run on a quadcore machine
Loading 20 newsgroups dataset forcategories
altatheism talkreligionmisc
1427 documents
2 categories
Performing grid search
pipeline vect tfidf clf
parameters
clfalpha 100000000000000001e05 99999999999999995e07
clfmmaxiter 10 50 80
clfpennalty l2 elasticnet
tfidfuseidf TrueFalse
521 Model Selection 1283
```


scikitlearn user guide Release 0213
vect CountVectorizer
tfidf TfidfTransformer
clf SGDClassifier

uncommenting more parameters will give better exploring power but will
increase processing time in a combinatorial way
parameters
vectmaxdf 05 075 10
vectmaxfeatures None 5000 10000 50000
vectngramrange 1 1 1 2 unigrams or bigrams
tfidfuseidf True False
tfidfnorm l1 l2
clfmaxiter 20
clfalpha 000001 0000001
clfpenalty l2 elasticnet
clfmaxiter 10 50 80

ifname main
multiprocessing requires the fork to happen in a main protected
block
find the best parameters for both the feature extraction and the
classifier
gridsearch GridSearchCVpipeline parameters cv5
njobs1 verbose1
printPerforming grid search
printpipeline name forname inpipelinesteps
printparameters
pprintparameters
t0 time
gridsearchfitdatadata datatarget
printdone in 03fs time t0
print
printBest score 03f gridsearchbestscore
printBest parameters set
bestparameters gridsearchbestestimatorgetparams
forparamname insortedparameterskeys
printtsr paramname bestparametersparamname
Total running time of the script 0 minutes 0000 seconds
Note Click here to download the full example code
52112 Confusion matrix
Example of confusion matrix usage to evaluate the quality of the output of a classifier on the iris data set The diagonal
elements represent the number of points for which the predicted label is equal to the true label while offdiagonal
elements are those that are mislabeled by the classifier The higher the diagonal values of the confusion matrix the
better indicating many correct predictions
The figures show the confusion matrix with and without normalization by class support size number of elements in
521 Model Selection 1285

scikitlearn user guide Release 0213
each class This kind of normalization can be interesting in case of class imbalance to have a more visual interpretation of which class is being misclassified
Here the results are not as good as they could be as our choice for the regularization parameter C was not the best In real life applications this parameter is usually chosen using Tuning the hyperparameters of an estimator

-

1286 Chapter 5 Examples

scikitlearn user guide Release 0213

•

Out

Confusion matrix without normalization

13 0 0
0 10 6
0 0 9

Normalized confusion matrix

1 0 0
0 062 038
0 0 1

printdoc

import numpy as np

import matplotlib.pyplot as plt

from sklearn import svm datasets

from sklearnmodelselection import traintestsplit

from sklearnmetrics import confusionmatrix

from sklearnutilsmulticlass import uniquelabels

import some data to play with

521 Model Selection 1287

```
scikitlearn user guide Release 0213
iris datasetsloadiris
X irisdata
y iristarget
classnames iristargetnames
Split the data into a training set and a test set
Xtrain Xtest ytrain ytest traintestsplitX y randomstate0
Run classifier using a model that is too regularized C too low to see
the impact on the results
classifier svmSVCKernellinear C001
ypred classifierfitXtrain ytrainpredictXtest
defplotconfusionmatrixxytrue ypred classes
normalizeFalse
titleNone
cmappltcmBlues
```

This function prints and plots the confusion matrix
Normalization can be applied by setting normalizeTrue

```
if nottitle
ifnormalize
title Normalized confusion matrix
else
title Confusion matrix without normalization
Compute confusion matrix
cm confusionmatrixxytrue ypred
Only use the labels that appear in the data
classes classesuniqueylabelsytrue ypred
ifnormalize
cm cmastypefloat cmsumaxis1 npnewaxis
printNormalized confusion matrix
else
printConfusion matrix without normalization
printcm
fig ax pltsubplots
im aximshowcm interpolationnearest cmapcmap
axfigurecolorbarim axax
We want to show all ticks
axsetxticksnparangebcmshape1
yticksnparangebcmshape0
and label them with the respective list entries
xticklabelsclasses yticklabelsclasses
titletitle
ylabelTrue label
xlabelPredicted label
Rotate the tick labels and set their alignment
pltsetpaxgetxticklabels rotation45 haright
rotationmodeanchor
Loop over data dimensions and create text annotations
fmt 2f ifnormalize elsed
1288 Chapter 5 Examples
```

scikitlearn user guide Release 0213

```
thresh cmmax 2
foriinrange(mshape0
forjijnrange(mshape1
axtextj i formatcmi j fmt
hacenter vacenter
colorwhite ifcmi j thresh elseblack
figtightlayout
returnax
npsetprintoptionsprecision2
Plot nonnormalized confusion matrix
plotconfusionmatrix(ytest ypred classesclassnames
titleConfusion matrix without normalization
Plot normalized confusion matrix
plotconfusionmatrix(ytest ypred classesclassnames normalizeTrue
titleNormalized confusion matrix
pltshow
```

Total running time of the script 0 minutes 0217 seconds

Note Click here to download the full example code

52113 Visualizing crossvalidation behavior in scikitlearn

Choosing the right crossvalidation object is a crucial part of fitting a model properly There are many ways to split data into training and test sets in order to avoid model overfitting to standardize the number of groups in test sets etc This example visualizes the behavior of several common scikitlearn objects for comparison

```
from sklearn.model_selection import TimeSeriesSplit KFold ShuffleSplit
StratifiedKFold GroupShuffleSplit
GroupKFold StratifiedShuffleSplit
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Patch
np.random.seed(1338)
cmapdata plt.cm.Paired
cmapcv plt.cm.coolwarm
nsplits 4
```

Visualize our data

First we must understand the structure of our data It has 100 randomly generated input datapoints 3 classes split unevenly across datapoints and 10 “groups” split evenly across datapoints As we’ll see some crossvalidation objects do specific things with labeled data others behave differently with grouped data and others do not use this information To begin we’ll visualize our data

521 Model Selection 1289

```
scikitlearn user guide Release 0213
Generate the classgroup data
npoints 100
X nprandomrandn100 10
percentilesclasses 1 3 6
y nphstackii int100 perc
forii perc inenumeratepercentilesclasses
    Evenly spaced groups repeated once
groups nphstackii 10foriiinrange10
defvisualizegroupsclasses groups name
    Visualize dataset groups
fig ax pltsubplots
axscatterrangelenngroups 5 lengroups cgroups marker
lw50 cmapcmapdata
axscatterrangelenngroups 35 lengroups cclasses marker
lw50 cmapcmapdata
axsetylim1 5 yticks5 35
yticklabelsData ngroup Data nclass xlabelSample index
visualizegroupsy groups no groups
1290 Chapter 5 Examples
```


scikitlearn user guide Release 0213

Define a function to visualize crossvalidation behavior

We'll define a function that lets us visualize the behavior of each crossvalidation object We'll perform 4 splits of the data On each split we'll visualize the indices chosen for the training set in blue and the test set in red

defplotcvindicescv X y group ax nsplits lw10

Create a sample plot for indices of a crossvalidation object

Generate the trainingtesting visualizations for each CV (Cross-Validation) split

forii tr tt inenumeratecvsplitXX yy groupsgroup

Fill in indices with the trainingtest groups

indices nparraynnpnan lenX

indicestt 1

indicestr 0

Visualize the results

axscatterrangelenindices ii 5 lenindices

cindices marker lwlw cmapcmapcv

vmin2 vmax12

Plot the data classes and groups at the end

axscatterrangelenX ii 15 lenX

cy marker lwlw cmapcmapdata

axscatterrangelenX ii 25 lenX

cgroup marker lwlw cmapcmapdata

Formatting

yticklabels listrangensplits class group

axsetyticksnparangensplits2 5 yticklabelsyticklabels

xlabelSample index ylabelCV iteration

ylimnsplits22 2 xlim0 100

axsettitleformattypecvname fontsize15

returnax

Let's see how it looks for the KFold crossvalidation object

fig ax pltsubplots

cv KFoldnsplits

plotcvindicescv X y groups ax nsplits

521 Model Selection 1291

scikitlearn user guide Release 0.21.3

As you can see by default the KFold crossvalidation iterator does not take either datapoint class or group into consideration We can change this by using the StratifiedKFold like so

```
fig, ax = plt.subplots()
cv = StratifiedKFold(n_splits=5)
plot_cv_indices(cv, X, y, groups=ax, nsplits=5)
```

scikitlearn user guide Release 0213

In this case the crossvalidation retained the same ratio of classes across each CV (Cross-Validation) split Next we'll visualize this behavior for a number of CV (Cross-Validation) iterators

Visualize crossvalidation indices for many CV (Cross-Validation) objects

Let's visually compare the cross validation behavior for many scikitlearn crossvalidation objects Below we will loop through several common crossvalidation objects visualizing the behavior of each

Note how some use the groupclass information while others do not

cvs KFold GroupKFold ShuffleSplit StratifiedKFold

GroupShuffleSplit StratifiedShuffleSplit TimeSeriesSplit

for cv in cvs:

 thiscv = cv

 fig, ax = plt.subplots(figsize=(6, 3))

 plot_cv_indices(thiscv, X, y, groups=ax, nsplits=10)

 ax.legend(patch_color_map=cv02, patch_color_map=cv02)

 Testing set Training set loc102 8

 Make the legend fit

 plt.tight_layout()

 fig.subplots_adjust(right=0.7)

plt.show()

521 Model Selection 1293

-
-

scikitlearn user guide Release 0213

-
-

521 Model Selection 1295

-
-

scikitlearn user guide Release 0213

• Total running time of the script 0 minutes 0401 seconds

Note Click here to download the full example code

52114 Receiver Operating Characteristic ROC

Example of Receiver Operating Characteristic ROC metric to evaluate classifier output quality

ROC curves typically feature true positive rate on the Y axis and false positive rate on the X axis This means that the top left corner of the plot is the “ideal” point a false positive rate of zero and a true positive rate of one This is not very realistic but it does mean that a larger area under the curve AUC is usually better

The “steepness” of ROC curves is also important since it is ideal to maximize the true positive rate while minimizing the false positive rate

Multiclass settings

ROC curves are typically used in binary classification to study the output of a classifier In order to extend ROC curve and ROC area to multiclass or multilabel classification it is necessary to binarize the output One ROC curve can be drawn per label but one can also draw a ROC curve by considering each element of the label indicator matrix as a binary prediction microaveraging

Another evaluation measure for multiclass classification is macroaveraging which gives equal weight to the classification of each label

Note

See also sklearnmetricsrocaucscore Receiver Operating Characteristic ROC with cross validation

printdoc

import numpy as np

521 Model Selection 1297

```
scikitlearn user guide Release 0213
import matplotlib.pyplot as plt
from itertools import cycle
from sklearn import svm datasets
from sklearnmetrics import roccurve auc
from sklearnmodelselection import traintestsplit
from sklearnpreprocessing import labelbinarize
from sklearnmulticlass import OneVsRestClassifier
from scipy import interp
    Import some data to play with
iris datasetsloadiris
X irisdata
y iristarget
    Binarize the output
y labelbinarize y classes0 1 2
nclasses yshape1
    Add noisy features to make the problem harder
randomstate np.random.RandomState0
nsamples nfeatures Xshape
X np.concatenate((X, randomstate.randn(nsamples, nfeatures - Xshape[1])))
    shuffle and split training and test sets
Xtrain Xtest ytrain ytest traintestsplit(X, y, testsize=5,
randomstate0)
    Learn to predict each class against the other
classifier OneVsRestClassifier(svm.SVC(kernel='linear', probability=True,
randomstate=randomstate))
yscore classifier.fit(Xtrain, ytrain).decision_function(Xtest)
    Compute ROC curve and ROC area for each class
fpr dict
tpr dict
rocauc dict
for i in range(nclasses):
    fpr[i], tpr[i], roccurve, test_i, yscore_i = roccurve(yscore[i], ytest[i],
rocauc[i], aucfpr[i], tpr[i])
    Compute microaverage ROC curve and ROC area
fpr_micro, tpr_micro, roccurve, test_micro, yscore_micro = roccurve(yscore, ytest,
rocauc_micro, aucfpr_micro, tpr_micro)
    Plot of a ROC curve for a specific class
plt.figure()
lw = 2
plt.plot(fpr2, tpr2, color='darkorange', lw=lw, label='ROC curve area = %2f' % rocauc2)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic example')
plt.legend(loc='lower right')
```



```
scikitlearn user guide Release 0213
pltshow
Plot ROC curves for the multiclass problem
Compute macroaverage ROC curve and ROC area
First aggregate all false positive rates
allfpr = np.unique(np.concatenate(fpr for i in range(nclasses)))
Then interpolate all ROC curves at this points
meantpr = np.zeros_like(allfpr)
for i in range(nclasses):
    meantpr = interp(allfpr, fpr[i], tpr[i])
Finally average it and compute AUC
macro_tpr = np.mean(meantpr)
macro_fpr = allfpr
macro_auc = auc(macro_fpr, macro_tpr)
Plot all ROC curves
plt.figure()
plt.plot(micro_fpr, micro_tpr)
labelmicroaverage ROC curve area = 0.02
formatmacro_auc
521 Model Selection 1299
```

```

scikitlearn user guide Release 0213
colordeeppink linestyle linewidth4
pltplotfprmacro tprmacro
labelmacroaverage ROC curve area 002f
formatrocaucmacro
colornavy linestyle linewidth4
colors cycleaqua darkorange cornflowerblue
fori color inziprangenclasses colors
pltplotfpri tpri colorcolor lwlw
labelROC curve of class 0 area 102f
formati rocauci
pltplot0 1 0 1 k lwlw
pltxlim00 10
pltylim00 105
pltxlabelFalse Positive Rate
pltylabelTrue Positive Rate
plttitleSome extension of Receiver operating characteristic to multiclass
pltlegendloclower right
pltshow
Total running time of the script 0 minutes 0145 seconds
1300 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Note [Click here to download the full example code](#)

52115 Plotting Learning Curves

On the left side the learning curve of a naive Bayes classifier is shown for the digits dataset Note that the training score and the crossvalidation score are both not very good at the end However the shape of the curve can be found in more complex datasets very often the training score is very high at the beginning and decreases and the cross validation score is very low at the beginning and increases On the right side we see the learning curve of an SVM (Support Vector Machine) with RBF (Radial Basis Function) kernel We can see clearly that the training score is still around the maximum and the validation score could be increased with more training samples

•
521 Model Selection 1301

scikitlearn user guide Release 0213

```
•
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearnnaivebayes import GaussianNB
from sklearnsvm import SVC (Support Vector Classification)
from sklearndatasets import loaddigits
from sklearnmodelselection import learningcurve
from sklearnmodelselection import ShuffleSplit
defplotlearningcurveestimator title X y ylimNone cvNone
njobsNone trainsizesnplinspace1 10 5
```

Generate a simple plot of the test and training learning curve
Parameters

estimator object type that implements the fit and predict methods
An object of that type which is cloned for each validation
title string
Title for the chart
X arraylike shape nsamples nfeatures
Training vector where nsamples is the number of samples and
nfeatures is the number of features
1302 Chapter 5 Examples

scikitlearn user guide Release 0213

y arraylike shape nsamples or nsamples nfeatures optional
Target relative to X for classification or regression
None for unsupervised learning

ylim tuple shape ymin ymax optional
Defines minimum and maximum yvalues plotted

cv int crossvalidation generator or an iterable optional
Determines the crossvalidation splitting strategy
Possible inputs for cv are
None to use the default 3fold crossvalidation
integer to specify the number of folds
termCV splitter
An iterable yielding train test splits as arrays of indices
For integerNone inputs if y is binary or multiclass
classStratifiedKFold used If the estimator is not a classifier
or if y is neither binary nor multiclass classKFold is used
Refer refUser Guide crossvalidation for the various
crossvalidators that can be used here

njobs int or None optional defaultNone
Number of jobs to run in parallel
None means 1 unless in a objjoblibparallelbackend context
1 means using all processors See termGlossary njobs
for more details

trainsizes arraylike shape nticks dtype float or int
Relative or absolute numbers of training examples that will be used to
generate the learning curve If the dtype is float it is regarded as a
fraction of the maximum size of the training set that is determined
by the selected validation method ie it has to be within 0 1
Otherwise it is interpreted as absolute sizes of the training sets
Note that for classification the number of samples usually have to
be big enough to contain at least one sample from each class
default nplinspace01 10 5

pltfigure
plttitletitle
ifylimis notNone
ptylim ylim
pltxlabelTraining examples
ptylabelScore
trainsizes trainscores testscores learningcurve
estimator X y cvcv njobsnjobs trainsizestrainsizes
trainscoresmean npmeantrainscores axis1
trainscoresstd npstdtrainscores axis1
testscoresmean npmeantestscores axis1
testscoresstd npstdtestscores axis1
pltgrid
pltfillbetweentrainsizes trainscoresmean trainscoresstd
trainscoresmean trainscoresstd alpha01
colorr
pltfillbetweentrainsizes testscoresmean testscoresstd
testscoresmean testscoresstd alpha01 colorg

521 Model Selection 1303

scikitlearn user guide Release 0213

plt.plot(train_sizes, train_scores, mean, color, label='Training score')

plt.plot(train_sizes, test_scores, mean, color, label='Crossvalidation score')

plt.legend(loc='best')

return plt

digits = load_digits

X, y = digits.data, digits.target

title = 'Learning Curves Naive Bayes'

Cross validation with 100 iterations to get smoother mean test and train

score curves each time with 20 data randomly selected as a validation set

cv = ShuffleSplit(n_splits=10, test_size=0.2, random_state=0)

estimator = GaussianNB

plot_learning_curve(estimator, title, X, y, ylim=(0.7, 1.0), cv=cv, n_jobs=4)

title = 'Learning Curves SVM (Support Vector Machine) RBF (Radial Basis Function) kernel gamma=0.001'

SVC (Support Vector Classification) is more expensive so we do a lower number of CV (Cross-Validation) iterations

cv = ShuffleSplit(n_splits=10, test_size=0.2, random_state=0)

estimator = SVC(gamma=0.001)

plot_learning_curve(estimator, title, X, y, ylim=(0.7, 1.0), cv=cv, n_jobs=4)

plt.show()

Total running time of the script: 0 minutes 28.56 seconds

Note: Click here to download the full example code

52116 PrecisionRecall

Example of PrecisionRecall metric to evaluate classifier output quality

PrecisionRecall is a useful measure of success of prediction when the classes are very imbalanced. In information

retrieval, precision is a measure of result relevancy while recall is a measure of how many truly relevant results are

returned.

The precision-recall curve shows the tradeoff between precision and recall for different thresholds. A high area under

the curve represents both high recall and high precision, where high precision relates to a low false positive rate and

high recall relates to a low false negative rate. High scores for both show that the classifier is returning accurate results.

High precision as well as returning a majority of all positive results (high recall).

A system with high recall but low precision returns many results but most of its predicted labels are incorrect when

compared to the training labels. A system with high precision but low recall is just the opposite, returning very few

results but most of its predicted labels are correct when compared to the training labels. An ideal system with high

precision and high recall will return many results with all results labeled correctly.

Precision is defined as the number of true positives over the number of true positives plus the number of false

positives.

1304

Chapter 5

Examples

Recall $\frac{TP}{TP+FN}$ is defined as the number of true positives TP over the number of true positives plus the number of false negatives FN

$\frac{TP}{TP+FN}$
 $\frac{TP}{TP+FN}$

These quantities are also related to the F_1 score which is defined as the harmonic mean of precision and recall

$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$
 F_1

Note that the precision may not decrease with recall The definition of precision $\frac{TP}{TP+FP}$

$\frac{TP}{TP+FP}$ shows that lowering the threshold of a classifier may increase the denominator by increasing the number of results returned If the threshold was previously set too high the new results may all be true positives which will increase precision If the previous threshold was about right or too low further lowering the threshold will introduce false positives decreasing precision Recall is defined as $\frac{TP}{TP+FN}$

$\frac{TP}{TP+FN}$ where FN does not depend on the classifier threshold This means that lowering the classifier threshold may increase recall by increasing the number of true positive results It is also possible that lowering the threshold may leave recall unchanged while the precision fluctuates The relationship between recall and precision can be observed in the stairstep area of the plot at the edges of these steps a small change in the threshold considerably reduces precision with only a minor gain in recall Average precision AP summarizes such a plot as the weighted mean of precisions achieved at each threshold with the increase in recall from the previous threshold used as the weight

$AP = \sum_{i=1}^n \frac{precision_i \times (recall_i - recall_{i-1})}{recall_i - recall_{i-1}}$

$precision_i \times (recall_i - recall_{i-1})$ where $precision_i$ and $recall_i$ are the precision and recall at the i th threshold A pair $(precision_i, recall_i)$ is referred to as an operating point AP and the trapezoidal area under the operating points `sklearn.metrics.auc` are common ways to summarize a precisionrecall curve that lead to different results Read more in the User Guide

Precisionrecall curves are typically used in binary classification to study the output of a classifier In order to extend the precisionrecall curve and average precision to multiclass or multilabel classification it is necessary to binarize the output One curve can be drawn per label but one can also draw a precisionrecall curve by considering each element of the label indicator matrix as a binary prediction microaveraging

Note

See also `sklearn.metrics.average_precision_score` `sklearn.metrics.recall_score` `sklearn.metrics.precision_score` `sklearn.metrics.f1_score`

In binary classification settings

Create simple data

Try to differentiate the two first classes of the iris data

```
from sklearn import svm datasets
```

```
from sklearn.model_selection import train_test_split
```

```
import numpy as np
```

```
iris = datasets.load_iris()
```

```
X = iris.data
```

```
y = iris.target
```

Add noisy features

```
random_state = np.random.RandomState(0)
```

```
n_samples, n_features = X.shape
```

```
scikitlearn user guide Release 0213
X = np.c_[X_randomstate,randn(nsamples, 200 - nfeatures)]
# Limit to the two first classes and split into training and test
X_train, X_test, y_train, y_test = train_test_split(X, y, 2, 2,
                                                    test_size=5,
                                                    random_state=random_state)
# Create a simple classifier
classifier = svm.LinearSVC(random_state=random_state)
classifier.fit(X_train, y_train)
y_score = classifier.decision_function(X_test)
# Compute the average precision score
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(y_test, y_score)
print('Average precision-recall score: %0.2f' % (average_precision))
# Out
Average precision-recall score: 0.88
# Plot the Precision-Recall curve
from sklearn.metrics import precision_recall_curve
import matplotlib.pyplot as plt
from inspect import signature
precision, recall, _ = precision_recall_curve(y_test, y_score)
# In matplotlib 1.5 plt.fill_between does not have a step argument
stepkwargs = {'step': 'post'}
if 'step' in signature(plt.fill_between).parameters:
    else:
plt.step(recall, precision, color='b', alpha=0.2,
         where='post')
plt.fill_between(recall, precision, alpha=0.2, color='b', stepkwargs=stepkwargs)
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.ylim([0.0, 1.05])
plt.xlim([0.0, 1.0])
plt.title('2-class Precision-Recall curve (AP=%0.2f)' % (average_precision))
average_precision
1306 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
In multilabel settings
Create multilabel data fit and predict
We create a multilabel dataset to illustrate the precisionrecall in multilabel settings
from sklearn.preprocessing import labelbinarize
Use labelbinarize to be multilabel like settings
Y = labelbinarize(y, classes=[0, 1, 2])
n_classes = Y.shape[1]
Split into training and test
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=5,
                                                    random_state=random_state)
We use OneVsRestClassifier for multilabel prediction
from sklearn.multiclass import OneVsRestClassifier
Run classifier
classifier = OneVsRestClassifier(svm.LinearSVC(random_state=random_state))
classifier.fit(X_train, Y_train)
y_score = classifier.decision_function(X_test)
521 Model Selection 1307
```

```
scikitlearn user guide Release 0213
The average precision score in multilabel settings
from sklearnmetrics import precisionrecallcurve
from sklearnmetrics import averageprecisionscore
For each class
precision dict
recall dict
averageprecision dict
foriinrangenclasses
precisioni recalli precisionrecallcurveYtest i
yscore i
averageprecisioni averageprecisionscoreYtest i yscore i
A microaverage quantifying score on all classes jointly
precisionmicro recallmicro precisionrecallcurveYtesttravel
yscoreravel
averageprecisionmicro averageprecisionscoreYtest yscore
averagemicro
printAverage precision score microaveraged over all classes 002f
formataverageprecisionmicro
Out
Average precision score microaveraged over all classes 043
Plot the microaveraged PrecisionRecall curve
pltfigure
pltsteprecallmicro precisionmicro colorb alpha02
wherepost
pltfillbetweenrecallmicro precisionmicro alpha02 colorb
stepkwargs
pltxlabelRecall
pltylabelPrecision
pltylim00 105
pltxlim00 10
plttitle
Average precision score microaveraged over all classes AP002f
formataverageprecisionmicro
1308 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
Plot PrecisionRecall curve for each class and isof1 curves
from itertools import cycle
    setup plot details
colors    cyclenavy turquoise darkorange cornflowerblue teal
pltfigurefigsize7 8
fscores    nplinspace02 08 num4
lines
labels
forfscore infscores
x    nplinspace001 1
y    fscore x 2x    fscore
l    pltplotxy 0 yy 0 colorgray alpha02
pltannotatef1001ffformatfscore xy09 y45    002
linesappendl
labelsappendisof1 curves
l    pltplotrecallmicro precisionmicro colorgold lw2
linesappendl
labelsappendmicroaverage Precisionrecall area    002f
formataverageprecisionmicro
fori color inziprangenclases colors
521 Model Selection 1309
```

scikitlearn user guide Release 0.21.3
l = plt.plot(recall_i, precision_i, color=color, lw=2)
lines.append(l)
labels.append(PrecisionRecall for class 0 area = 102f)
format = averageprecision
fig = plt.gcf()
fig.subplots_adjust(bottom=0.25)
plt.xlim(0, 1.0)
plt.ylim(0, 1.05)
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.title('Extension of PrecisionRecall curve to multiclass')
plt.legend(lines, labels, loc=0, propdict={'size': 14})
plt.show()

1310 Chapter 5 Examples

scikitlearn user guide Release 0213
Total running time of the script 0 minutes 0064 seconds
522 Multioutput methods
Examples concerning the sklearnmultioutput module
522 Multioutput methods 1311

scikitlearn user guide Release 0213

Note Click here to download the full example code

5221 Classifier Chain

Example of using classifier chain on a multilabel dataset

For this example we will use the yeast dataset which contains 2417 datapoints each with 103 features and 14 possible labels Each data point has at least one label As a baseline we first train a logistic regression classifier for each of the 14 labels To evaluate the performance of these classifiers we predict on a heldout test set and calculate the jaccard score for each sample

Next we create 10 classifier chains Each classifier chain contains a logistic regression model for each of the 14 labels

The models in each chain are ordered randomly In addition to the 103 features in the dataset each model gets the predictions of the preceding models in the chain as features note that by default at training time each model gets the true labels as features These additional features allow each chain to exploit correlations among the classes The Jaccard similarity score for each chain tends to be greater than that of the set independent logistic models

Because the models in each chain are arranged randomly there is significant variation in performance among the chains Presumably there is an optimal ordering of the classes in a chain that will yield the best performance However we do not know that ordering a priori Instead we can construct an voting ensemble of classifier chains by averaging the binary predictions of the chains and apply a threshold of 05 The Jaccard similarity score of the ensemble is greater than that of the independent models and tends to exceed the score of each chain in the ensemble although this is not guaranteed with randomly ordered chains

Author Adam Kleczewski

License BSD (Binary Search Decision) 3 clause

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import fetchopenml

1312 Chapter 5 Examples

```
scikitlearn user guide Release 0213
from sklearnmultioutput import ClassifierChain
from sklearnmodelselection import traintestsplit
from sklearnmulticlass import OneVsRestClassifier
from sklearnmetrics import jaccardscore
from sklearnlinearmodel import LogisticRegression
printdoc
    Load a multilabel dataset from httpswwwopenmlorgd40597
X Y fetchopenmlyeast version4 returnXyTrue
Y Y TRUE
Xtrain Xtest Ytrain Ytest traintestsplitX Y testsize2
randomstate0
    Fit an independent logistic regression model for each class using the
    OneVsRestClassifier wrapper
baselr LogisticRegressionsoverlbfgs
ovr OneVsRestClassifierbaselr
ovrfitXtrain Ytrain
Ypredovr ovrpredictXtest
ovrjaccardscore jaccardscoreYtest Ypredovr averagesamples
    Fit an ensemble of logistic regression classifier chains and take the
    take the average prediction of all the chains
chains ClassifierChainbaselr orderrandom randomstatei
foriinrange10
forchaininchains
chainfitXtrain Ytrain
Ypredchains nparraychainpredictXtest forchainin
chains
chainjaccardscores jaccardscoreYtest Ypredchain 5
averagesamples
forYpredchain inYpredchains
Ypredensemble Ypredchainsmeanaxis0
ensemblejaccardscore jaccardscoreYtest
Ypredensemble 5
averagesamples
modelscores ovrjaccardscore chainjaccardscores
modelscoresappendensemblejaccardscore
modelnames Independent
Chain 1
Chain 2
Chain 3
Chain 4
Chain 5
Chain 6
Chain 7
Chain 8
Chain 9
Chain 10
Ensemble
xpos nparangelenmodelnames
522 Multioutput methods 1313
```

scikitlearn user guide Release 0213

Plot the Jaccard similarity scores for the independent model each of the chains and the ensemble note that the vertical axis on this plot does not begin at 0

```
fig, ax = plt.subplots(figsize=(7, 4))
```

```
ax.grid(True)
```

```
ax.set_title('Classifier Chain Ensemble Performance Comparison')
```

```
ax.set_xticks(xpos)
```

```
ax.set_xticklabels(model_names, rotation=vertical)
```

```
ax.set_ylabel('Jaccard Similarity Score')
```

```
ax.set_ylim(min(scores), max(scores))
```

```
colors = ['r', 'b', 'g']
```

```
ax.bar(xpos, model_scores, alpha=0.5, color=colors)
```

```
plt.tight_layout()
```

```
plt.show()
```

Total running time of the script: 0 minutes 79.47 seconds

523 Nearest Neighbors

Examples concerning the sklearn.neighbors module

Note: Click here to download the full example code

5231 Nearest Neighbors regression

Demonstrate the resolution of a regression problem using a kNearest Neighbor and the interpolation of the target using both barycenter and constant weights

1314 Chapter 5 Examples

scikitlearn user guide Release 0213
printdoc
Author Alexandre Gramfort alexandregramfortinriafr
Fabian Pedregosa fabianpedregosainriafr

License BSD (Binary Search Decision) 3 clause C INRIA

Generate sample data
import numpy as np
import matplotlib.pyplot as plt
from sklearn import neighbors
np.random.seed(0)
X = np.sort(5 * np.random.rand(40, 1), axis=0)
T = np.linspace(0, 5, 500).newaxis
y = np.sin(X.ravel())
Add noise to targets
y[5:10] = 0.5 * np.random.rand(5)

Fit regression model
n_neighbors = 5
523 Nearest Neighbors 1315

```
scikitlearn user guide Release 0.21.3
for i, weights in enumerate(uniform_distances):
    knn_neighbors = KNeighborsRegressor(n_neighbors=weights)
    y_knn_fit = knn_fit.predict(X_test)
    plt.subplot(2, 1, 1)
    plt.scatter(X_test, y_knn_fit, label='data')
    plt.plot(X_test, y_knn_fit, label='prediction')
    plt.axis('tight')
    plt.legend()
    plt.title('KNeighborsRegressor with n_neighbors = %s' % n_neighbors)
    weights.append(weights)
plt.tight_layout()
plt.show()
Total running time of the script: 0 minutes 0.089 seconds
Note: Click here to download the full example code
5.2.3.2 Outlier detection with Local Outlier Factor (LOF)
The Local Outlier Factor (LOF) algorithm is an unsupervised anomaly detection method which computes the local density deviation of a given data point with respect to its neighbors. It considers as outliers the samples that have a substantially lower density than their neighbors. This example shows how to use LOF for outlier detection which is the default use case of this estimator in scikitlearn. Note that when LOF is used for outlier detection it has no predict, decision_function and score_samples methods. See User Guide for details on the difference between outlier detection and novelty detection and how to use LOF for novelty detection.
The number of neighbors considered parameter n_neighbors is typically set 1 greater than the minimum number of samples a cluster has to contain so that other samples can be local outliers relative to this cluster and 2 smaller than the maximum number of close by samples that can potentially be local outliers. In practice such informations are generally not available and taking n_neighbors=20 appears to work well in general.
1316 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
import numpy as np
import matplotlib.pyplot as plt
from sklearn.neighbors import LocalOutlierFactor
printdoc
nprandomseed42
Generate train data
Xinliers 03 nprandomrandn100 2
Xinliers nprXinliers 2 Xinliers 2
Generate some outliers
Xoutliers nprandomuniformlow4 high4 size20 2
X nprXinliers Xoutliers
noutliers lenXoutliers
groundtruth np.oneslenX dtypeint
groundtruthnoutliers 1
fit the model for outlier detection default
clf LocalOutlierFactornneighbors20 contamination01
use fitpredict to compute the predicted labels of the training samples
when LOF is used for outlier detection the estimator has no predict
decisionfunction and scoresamples methods
ypred clffitpredictX
523 Nearest Neighbors 1317
```

scikitlearn user guide Release 0213

nerrors ypred groundtruthsum

Xscores clfnegativeoutlierfactor

plttitleLocal Outlier Factor LOF

pltscatterX 0 X 1 colork s3 labelData points

plot circles with radius proportional to the outlier scores

radius Xscoresmax Xscores Xscoresmax Xscoresmin

pltscatterX 0 X 1 s1000 radius edgecolorsr

facecolorsnone labelOutlier scores

pltaxistight

pltxlim5 5

pltylim5 5

pltxlabelprediction errors d nerrors

legend pltlegendlocupper left

legendlegendHandles0sizes 10

legendlegendHandles1sizes 20

pltshow

Total running time of the script 0 minutes 0017 seconds

Note [Click here to download the full example code](#)

5233 Nearest Neighbors Classification

Sample usage of Nearest Neighbors classification It will plot the decision boundaries for each class

1318 Chapter 5 Examples

scikitlearn user guide Release 0213

•

523 Nearest Neighbors 1319

scikitlearn user guide Release 0213

```
•
printdoc
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn import neighbors datasets
nneighbors = 15
import some data to play with
iris = datasets.load_iris
# we only take the first two features We could avoid this ugly
# slicing by using a twodim dataset
X = iris.data[:, 0:2]
y = iris.target
h = 0.2 # step size in the mesh
# Create color maps
cmap_light = ListedColormap(['#FFAAAA', '#AAFFAA', '#AAAAFF'])
cmap_bold = ListedColormap(['#FF0000', '#00FF00', '#0000FF'])
for weights in ['uniform', 'distance']:
    # we create an instance of Neighbours Classifier and fit the data
    clf = neighbors.KNeighborsClassifier(nneighbors, weights=weights)
    clf.fit(X, y)
```

scikitlearn user guide Release 0213

Plot the decision boundary For that we will assign a color to each point in the mesh xmin xmax ymin ymax

xmin xmax X 0min 1 X 0max 1

ymin ymax X 1min 1 X 1max 1

xx yy npmeshgridnparangexmin xmax h

nparangeymin ymax h

Z clfpredictnpcxxravel yy.ravel

Put the result into a color plot

Z Zreshapexxshape

pltfigure

pltcolormeshxx yy Z cmapcmaplight

Plot also the training points

pltscatterX 0 X 1 cy cmapcmapbold

edgecolor k s20

pltxlimxmin xxmax

pltylimymin ymax

plttitle3Class classification k i weights s

nneighbors weights

pltshow

Total running time of the script 0 minutes 1416 seconds

Note Click here to download the full example code

5234 Nearest Centroid Classification

Sample usage of Nearest Centroid classification It will plot the decision boundaries for each class

523 Nearest Neighbors 1321

scikitlearn user guide Release 0213

```
•
Out
None 08133333333333334
02 082
printdoc
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn import datasets
from sklearn.neighbors import NearestCentroid
nneighbors = 15
import some data to play with
iris = datasets.load_iris()
# we only take the first two features We could avoid this ugly
# slicing by using a twodim dataset
X = iris.data[:, 2:]
y = iris.target
523 Nearest Neighbors 1323
```

```
scikitlearn user guide Release 0213
h 02 step size in the mesh
Create color maps
cmaplight ListedColormapFFAAAA AAFFAA AAAAFF
cmapbold ListedColormapFF0000 00FF00 0000FF
forshrinkage inNone 2
we create an instance of Neighbours Classifier and fit the data
clf NearestCentroidshrinkthresholdshrinkage
clffitX y
ypred clfpredictX
printshrinkage npmean ypred
Plot the decision boundary For that we will assign a color to each
point in the mesh xmin xmaxxymin ymax
xmin xmax X 0min 1 X 0max 1
ymin ymax X 1min 1 X 1max 1
xx yy npmeshgridnparangexmin xmax h
nparangeymin ymax h
Z clfpredictnpcxxravel yy.ravel
Put the result into a color plot
Z Zreshapexxshape
pltfigure
pltcolormeshxx yy Z cmapcmaplight
Plot also the training points
pltscatterX 0 X 1 cy cmapcmapbold
edgecolor k s20
plttitle3Class classification shrinkthreshold r
shrinkage
pltaxistight
pltshow
Total running time of the script 0 minutes 0051 seconds
Note Click here to download the full example code
5235 Kernel Density Estimation
This example shows how kernel density estimation KDE a powerful nonparametric density estimation technique
can be used to learn a generative model for a dataset With this generative model in place new samples can be drawn
These new samples reflect the underlying model of the data
1324 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
Out
best bandwidth 379269019073225
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_digits
from sklearn.neighbors import KernelDensity
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.model_selection import GridSearchCV

load the data
digits = load_digits()
project the 64dimensional data to a lower dimension
pca = PCA(n_components=15, whiten=False)
data = pca.fit_transform(digits.data)
use grid search crossvalidation to optimize the bandwidth
params = {'bandwidth': np.logspace(1, 1, 20)}
523 Nearest Neighbors 1325
```

```
scikitlearn user guide Release 0213
grid GridSearchCVKernelDensity params cv5 iidFalse
gridfitdata
printbest bandwidth 0formatgridbestestimatorbandwidth
use the best estimator to compute the kernel density estimate
kde gridbestestimator
sample 44 new points from the data
newdata kdesample44 randomstate0
newdata pcainversetransformnewdata
turn data into a 4x11 grid
newdata newdatareshape4 11 1
realdata digitsdata44reshape4 11 1
plot real digits and resampled digits
fig ax pltsubplots9 11 subplotkwdictxticks yticks
forjinrange11
ax4 jsetvisibleFalse
foriinrange4
im axi jimshowrealdatai jreshape8 8
cmapppltcmbinary interpolationnearest
imsetclim0 16
im axi 5 jimshownewdatai jreshape8 8
cmapppltcmbinary interpolationnearest
imsetclim0 16
ax0 5settitleSelection from the input data
ax5 5settitleNew digits drawn from the kernel density model
pltshow
```

Total running time of the script 0 minutes 4482 seconds

Note Click here to download the full example code

5236 Novelty detection with Local Outlier Factor LOF

The Local Outlier Factor LOF algorithm is an unsupervised anomaly detection method which computes the local density deviation of a given data point with respect to its neighbors It considers as outliers the samples that have a substantially lower density than their neighbors This example shows how to use LOF for novelty detection Note that when LOF is used for novelty detection you MUST not use predict decisionfunction and scoresamples on the training set as this would lead to wrong results You must only use these methods on new unseen data which are not in the training set See User Guide for details on the difference between outlier detection and novelty detection and how to use LOF for outlier detection

The number of neighbors considered parameter nneighbors is typically set 1 greater than the minimum number of samples a cluster has to contain so that other samples can be local outliers relative to this cluster and 2 smaller than the maximum number of close by samples that can potentially be local outliers In practice such informations are generally not available and taking nneighbors20 appears to work well in general

```
scikitlearn user guide Release 0213
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from sklearn.neighbors import LocalOutlierFactor
printdoc
np.random.seed(42)
xx, yy = np.meshgrid(np.linspace(5, 5, 500), np.linspace(5, 5, 500))
# Generate normal not abnormal training observations
X = 0.3 * np.random.randn(100, 2)
X_train = np.r_[X, X, X, X, X, X, X, X, X, X]
# Generate new normal not abnormal observations
X = 0.3 * np.random.randn(20, 2)
X_test = np.r_[X, X, X, X, X, X, X, X, X, X]
# Generate some abnormal novel observations
X_outliers = np.random.uniform(low=-4, high=4, size=(20, 2))
# fit the model for novelty detection
novelty = True
clf = LocalOutlierFactor(n_neighbors=20, novelty=True, contamination=0.1)
clf.fit(X_train)
# DO NOT use predict, decision_function and score_samples on X_train as this
# would give wrong results but only on new unseen data not used in X_train
# eg X_test, X_outliers or the meshgrid
y_pred_test = clf.predict(X_test)
523 Nearest Neighbors 1327
```

```
scikitlearn user guide Release 0213
ypredoutliers clfpredictXoutliers
nerrortest ypredtestypredtest 1size
nerroroutliers ypredoutliersypredoutliers 1size
plot the learned frontier the points and the nearest vectors to the plane
Z clfdecisionfunctionnpcxxravel yyravel
Z Zreshapexxshape
plttitleNovelty Detection with LOF
pltcontourfxx yy Z levelsnplinspaceZmin 0 7 cmappltcmPuBu
a pltcontourxx yy Z levels0 linewidths2 colorsdarkred
pltcontourfxx yy Z levels0 Zmax colorspalevioletred
s 40
b1 pltscatterXtrain 0 Xtrain 1 cwhite ss edgecolorsk
b2 pltscatterXtest 0 Xtest 1 cblueviolet ss
edgecolorsk
c pltscatterXoutliers 0 Xoutliers 1 cgold ss
edgecolorsk
pltaxistight
pltxlim5 5
pltylim5 5
pltlegendacollections0 b1 b2 c
learned frontier training observations
new regular observations new abnormal observations
locupper left
propmatplotlibfontmanagerFontPropertiessize11
pltxlabel
errors novel regular d40 errors novel abnormal d40
nerrortest nerroroutliers
pltshow
Total running time of the script 0 minutes 0634 seconds
Note Click here to download the full example code
5237 Comparing Nearest Neighbors with and without Neighborhood Components
Analysis
An example comparing nearest neighbors classification with and without Neighborhood Components Analysis
It will plot the class decision boundaries given by a Nearest Neighbors classifier when using the Euclidean distance
on the original features versus using the Euclidean distance after the transformation learned by Neighborhood Com
ponents Analysis The latter aims to find a linear transformation that maximises the stochastic nearest neighbor
classification accuracy on the training set
1328 Chapter 5 Examples
```

scikitlearn user guide Release 0213

•

523 Nearest Neighbors 1329

scikitlearn user guide Release 0213

```
•
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
NeighborhoodComponentsAnalysis
from sklearn.pipeline import Pipeline
printdoc
nneighbors 1
dataset datasetsloadiris
X y datasetdata datasettarget
we only take two features We could avoid this ugly
slicing by using a twodim dataset
X X 0 2
Xtrain Xtest ytrain ytest
train_test_split(X y stratify y test_size=0.7 random_state=42
1330 Chapter 5 Examples
```


scikitlearn user guide Release 0213

h 01 step size in the mesh

Create color maps

cmaplight ListedColormapFFAAAA AAFFAA AAAAFF

cmapbold ListedColormapFF0000 00FF00 0000FF

names KNN NCA KNN

classifiers Pipelinescaler StandardScaler

knn KNeighborsClassifiernneighborsnneighbors

Pipelinescaler StandardScaler

nca NeighborhoodComponentsAnalysis

knn KNeighborsClassifiernneighborsnneighbors

xmin xmax X 0min 1 X 0max 1

ymin ymax X 1min 1 X 1max 1

xx yy npmeshgridnparangexmin xmax h

nparangeymin ymax h

forname clf inzipnames classifiers

clffitXtrain ytrain

score clfscoreXtest ytest

Plot the decision boundary For that we will assign a color to each

point in the mesh xmin xmaxxymin ymax

Z clfpredictnpcxxravel yy.ravel

Put the result into a color plot

Z Zreshapexxshape

pltfigure

pltcolormeshxx yy Z cmapcmaplight alpha8

Plot also the training and testing points

pltscatterX 0 X 1 cy cmapcmapbold edgecolor k s20

pltxlimxxmin xxmax

pltylimyymin yymax

plttitle k formatname nneighbors

plttext09 01 2fformatscore size15

hacenter vacenter transformpltgcatransAxes

pltshow

Total running time of the script 0 minutes 16006 seconds

Note Click here to download the full example code

5238 Dimensionality Reduction with Neighborhood Components Analysis

Sample usage of Neighborhood Components Analysis for dimensionality reduction

This example compares different linear dimensionality reduction methods applied on the Digits data set The data

523 Nearest Neighbors 1331

scikitlearn user guide Release 0213

set contains images of digits from 0 to 9 with approximately 180 samples of each class Each image is of dimension 8x8 64 and is reduced to a twodimensional data point

Principal Component Analysis PCA (Principal Component Analysis) applied to this data identifies the combination of attributes principal components or directions in the feature space that account for the most variance in the data Here we plot the different samples on the 2 first principal components

Linear Discriminant Analysis LDA tries to identify attributes that account for the most variance between classes In particular LDA in contrast to PCA (Principal Component Analysis) is a supervised method using known class labels

Neighborhood Components Analysis NCA tries to find a feature space such that a stochastic nearest neighbor algorithm will give the best accuracy Like LDA it is a supervised method

One can see that NCA enforces a clustering of the data that is visually meaningful despite the large reduction in dimension

•

1332 Chapter 5 Examples

scikitlearn user guide Release 0213

- 523 Nearest Neighbors 1333

scikitlearn user guide Release 0213

```
•
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.neighbors import KNeighborsClassifier
NeighborhoodComponentsAnalysis
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
printdoc
nneighbors 3
randomstate 0
Load Digits dataset
digits = datasets.load_digits
X y = digits.data digitstarget
Split into train test
Xtrain Xtest ytrain ytest
train_test_split(X y testsize=0.5 stratify=y
randomstate=randomstate
1334 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
dim lenX0
nclasses lennpuniquey
  Reduce dimension to 2 with PCA (Principal Component Analysis)
pca makepipelineStandardScaler
PCAncomponents2 randomstaterandomstate
  Reduce dimension to 2 with LinearDiscriminantAnalysis
lda makepipelineStandardScaler
LinearDiscriminantAnalysisncomponents2
  Reduce dimension to 2 with NeighborhoodComponentAnalysis
nca makepipelineStandardScaler
NeighborhoodComponentsAnalysisncomponents2
randomstaterandomstate
  Use a nearest neighbor classifier to evaluate the methods
knn KNeighborsClassifiernneighborsnneighbors
  Make a list of the methods to be compared
dimreductionmethods PCA (Principal Component Analysis) pca LDA lda NCA nca
pltfigure
fori name model inenumeratedimreductionmethods
pltfigure
  pltsubplot1 3 i 1 aspect1
  Fit the methods model
modelfitXtrain ytrain
  Fit a nearest neighbor classifier on the embedded training set
knnfitmodeltransformXtrain ytrain
  Compute the nearest neighbor accuracy on the embedded test set
accknn knnscoremodeltransformXtest ytest
  Embed the data set in 2 dimensions using the fitted model
Xembedded modeltransformX
  Plot the projected points and show the evaluation score
pltscatterXembedded 0 Xembedded 1 cy s30 cmapSet1
plttitle KNN k nTest accuracy 2fformatname
nneighbors
accknn
pltshow
Total running time of the script 0 minutes 2879 seconds
Note Click here to download the full example code
5239 Kernel Density Estimate of Species Distributions
This shows an example of a neighborsbased query in particular a kernel density estimate on geospatial data using
a Ball Tree built upon the Haversine distance metric - ie distances over points in latitudelongitude The dataset
523 Nearest Neighbors 1335
```

scikitlearn user guide Release 0213

is provided by Phillips et al 2006 If available the example uses basemap to plot the coast lines and national boundaries of South America

This example does not perform any learning over the data see Species distribution modeling for an example of classification based on the attributes in this dataset It simply shows the kernel density estimate of observed data points in geospatial coordinates

The two species are

- “Bradypus variegatus” the Brownthroated Sloth
- “Microryzomys minutus” also known as the Forest Small Rice Rat a rodent that lives in Peru Colombia Ecuador Peru and Venezuela

References

- “Maximum entropy modeling of species geographic distributions” S J Phillips R P Anderson R E Schapire Ecological Modelling 190231259 2006

Out

computing KDE in spherical coordinates

plot coastlines from coverage

computing KDE in spherical coordinates

plot coastlines from coverage

1336 Chapter 5 Examples

scikitlearn user guide Release 0213
Author Jake Vanderplas jakevdp@cs.washington.edu

```
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_species_distributions
from sklearn.datasets.species_distributions import construct_grids
from sklearn.neighbors import KernelDensity
if basemap is available we'll use it
    otherwise we'll improvise later
try
    from mpl_toolkits.basemap import Basemap
    basemap = True
except ImportError:
    basemap = False
    Get matrices/arrays of species IDs and locations
    data = fetch_species_distributions(
        species_names='Bradypus Variegatus Microryzomys Minutus',
        X_train=np.vstack(data['train']['lat',
                               'train']['long']),
        y_train=np.array(data['train']['species']),
        for_in_data=train['species'], dtype=int)
    X_train = np.pi * 180  # Convert lat/long to radians
    Set up the data grid for the contour plot
    xgrid, ygrid = construct_grids(data)
    X, Y = np.meshgrid(xgrid, ygrid)
    landreference = data['coverages']
    landmask = landreference % 9999
    xy = np.vstack(Y.ravel(), X.ravel())
    xy = xy[landmask]
    xy = xy[0:180]
    Plot map of South America with distributions of each species
    fig = plt.figure()
    fig.subplots_adjust(left=0.05, right=0.95, wspace=0.05)
    for i in range(2):
        plt.subplot(2, 2, i + 1)
        Construct a kernel density estimate of the distribution
        print 'Computing KDE in spherical coordinates'
        kde = KernelDensity(bandwidth=0.04, metric='haversine',
                              kernel='gaussian', algorithm='balltree')
        kde.fit(X_train, y_train)
        Evaluate only on the land 9999 indicates ocean
        Z = np.full(landmask.shape, 9999, dtype=int)
        Z[landmask] = np.exp(kde.score_samples(xy))
        Z = Z.reshape(X.shape)
    523 Nearest Neighbors 1337
```

```
scikitlearn user guide Release 0213
plot contours of the density
levels nplinspace0 Zmax 25
pltcontourfX Y Z levelslevels cmappltcmReds
ifbasemap
print plot coastlines using basemap
m Basemapprojectioncyl llcrnrlatYmin
urcrnrlatYmax llcrnrlonXmin
urcrnrlonXmax resolutionc
mdrawcoastlines
mdrawcountries
else
print plot coastlines from coverage
pltcontourX Y landreference
levels9998 colorsk
linestylesolid
pltxticks
pltyticks
plttitlespeciesnamesi
pltshow
Total running time of the script 0 minutes 5470 seconds
Note Click here to download the full example code
52310 Neighborhood Components Analysis Illustration
An example illustrating the goal of learning a distance metric that maximizes the nearest neighbors classification
accuracy The example is solely for illustration purposes Please refer to the User Guide for more information
1338 Chapter 5 Examples
```


scikitlearn user guide Release 0213

•

523 Nearest Neighbors 1339

scikitlearn user guide Release 0213

```
•
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.neighbors import NeighborhoodComponentsAnalysis
from matplotlib import cm
from sklearnutilsfixes import logsumexp
printdoc
nneighbors 1
randomstate 0
Create a tiny data set of 9 samples from 3 classes
X y make_classification(nsamples9 nfeatures2 ninformative2
nredundant0 nclasses3 nclustersperclass1
classep10 randomstaterandomstate
Plot the points in the original space
pltfigure
ax pltgca
Draw the graph nodes
foriinrangeXshape0
axtextXi 0 Xi 1 stri vacenter hacenter
axscatterXi 0 Xi 1 s300 ccmSet1yi alpha04
1340 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
def piX i
    diffembedded Xi X
    distembedded npeinsumijiji diffembedded
    diffembedded
    distembedded i npinf
        compute exponentiated distances use the logsumexp trick to
        avoid numerical instabilities
    expdistembedded npexpdistembedded
    logsumexpdistembedded
    returnexpdistembedded
    defrelatepointX i ax
    pti Xi
    forj ptj in enumerateX
    thickness piX i
    ifi j
    line pti0 ptj0 pti1 ptj1
    axplot line ccmSet1yj
    linewidth5 thicknessj
    we consider only point 3
    i 3
    Plot bonds linked to sample i in the original space
    relatepointX i ax
    axsettitleOriginal points
    axesgetxaxissetvisibleFalse
    axesgetyaxissetvisibleFalse
    axesequal
    Learn an embedding with NeighborhoodComponentsAnalysis
    nca NeighborhoodComponentsAnalysismaxiter30 randomstaterandomstate
    nca ncafitX y
    Plot the points after transformation with NeighborhoodComponentsAnalysis
    pltfigure
    ax2 pltgca
    Get the embedding and find the new nearest neighbors
    Xembedded ncatransformX
    relatepointXembedded i ax2
    foriinrangelenX
    ax2textXembedded i 0 Xembedded i 1 stri
    vacenter hacenter
    ax2scatterXembedded i 0 Xembedded i 1 s300 ccmSet1yi
    alpha04
    Make axes equal so that boundaries are displayed correctly as circles
    ax2settitleNCA embedding
    ax2axesgetxaxissetvisibleFalse
    ax2axesgetyaxissetvisibleFalse
    523 Nearest Neighbors 1341
```

scikitlearn user guide Release 0213

ax2axisequal

pltshow

Total running time of the script 0 minutes 0069 seconds

Note Click here to download the full example code

52311 Simple 1D Kernel Density Estimation

This example uses the `sklearn.neighbors.KernelDensity` class to demonstrate the principles of Kernel Density Estimation in one dimension

The first plot shows one of the problems with using histograms to visualize the density of points in 1D Intuitively a histogram can be thought of as a scheme in which a unit “block” is stacked above each point on a regular grid As the top two panels show however the choice of gridding for these blocks can lead to wildly divergent ideas about the underlying shape of the density distribution If we instead center each block on the point it represents we get the estimate shown in the bottom left panel This is a kernel density estimation with a “top hat” kernel This idea can be generalized to other kernel shapes the bottomright panel of the first figure shows a Gaussian kernel density estimate over the same distribution

Scikitlearn implements efficient kernel density estimation using either a Ball Tree or KD Tree structure through the `sklearn.neighbors.KernelDensity` estimator The available kernels are shown in the second figure of this example

The third figure compares kernel density estimates for a distribution of 100 samples in 1 dimension Though this example uses 1D distributions kernel density estimation is easily and efficiently extensible to higher dimensions as well

1342 Chapter 5 Examples

scikitlearn user guide Release 0213

•

523 Nearest Neighbors 1343

scikitlearn user guide Release 0213

- Author Jake Vanderplas jakevdp@cs.washington.edu

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from distutils.version import LooseVersion
from scipy.stats import norm
from sklearn.neighbors import KernelDensity
    normed is being deprecated in favor of density in histograms
if LooseVersion(matplotlib.__version__) < 2.1:
    densityparam = density = True
else:
    densityparam = normed = True
```

```
Plot the progression of histograms to kernels
np.random.seed(1)
N = 20
X = np.concatenate(np.random.normal(0, 1, int(0.3 * N)),
                    np.random.normal(5, 1, int(0.7 * N)))
np.random.shuffle(X)
Xplot = X[np.linspace(0, 1, 1000)]
bins = np.linspace(0, 1, 10)
fig, ax = plt.subplots(2, 2, sharex=True, sharey=True)
fig.subplots_adjust(hspace=0.05, wspace=0.05)
5.2.3 Nearest Neighbors 1345
```

```
scikitlearn user guide Release 0213
    histogram 1
ax0 0histX 0 binsbins fcAAAAFF densityparam
ax0 0text35 031 Histogram
    histogram 2
ax0 1histX 0 binsbins 075 fcAAAAFF densityparam
ax0 1text35 031 Histogram bins shifted
    tophat KDE
kde KernelDensitykerneltophat bandwidth075fitX
logdens kdescoresamplesXplot
ax1 0fillXplot 0 npexplogdens fcAAAAFF
ax1 0text35 031 Tophat Kernel Density
    Gaussian KDE
kde KernelDensitykernelgaussian bandwidth075fitX
logdens kdescoresamplesXplot
ax1 1fillXplot 0 npexplogdens fcAAAAFF
ax1 1text35 031 Gaussian Kernel Density
foraxiinaxravel
axiplotX 0 npfullXshape0 001 k
axisetxlim4 9
axisetylim002 034
foraxiinax 0
axisetylabelNormalized Density
foraxiinax1
axisetxlabelx

    Plot all available kernels
Xplot nplinspace6 6 1000 None
Xsrc npzeros1 1
fig ax pltsubplots2 3 sharexTrue shareyTrue
figsubplotsadjustleft005 right095 hspace005 wspace005
defformatfuncx loc
ifx 0
return0
elifx 1
returnh
elifx 1
returnh
else
returnih x
fori kernel inenumerategaussian tophat epanechnikov
exponential linear cosine
axi axraveli
logdens KernelDensitykernelkernelfitXsrcscoresamplesXplot
axifillXplot 0 npexplogdens k fcAAAAFF
axitext26 095 kernel
axixaxissetmajorformatterpltFuncFormatterformatfunc
1346 Chapter 5 Examples
```


scikitlearn user guide Release 0213
axixaxissetmajorlocatorpltMultipleLocator1
axiyaxissetmajorlocatorpltNullLocator
axisetylim0 105
axisetxlim29 29
ax0 1settitleAvailable Kernels

Plot a 1D density example
N 100
np.random.seed(1)
X = np.concatenate(np.random.normal(0, 1, int(0.3 * N)),
np.random.normal(5, 1, int(0.7 * N)).reshape(-1))
Xplot = np.linspace(0, 10, 1000).reshape(-1)
true_dens = 0.3 * norm.pdf(Xplot, 0, 1) + 0.7 * norm.pdf(Xplot, 5, 1)
fig, ax = plt.subplots(1, 1)
ax.fill(Xplot, true_dens, fc='black', alpha=0.2)
label = input('distribution: ')
for kernel in ['gaussian', 'tophat', 'epanechnikov', 'kde', 'KernelDensity', 'kernel', 'kernel', 'bandwidth05', 'fitX', 'logdens', 'kdescoresamples', 'Xplot', 'axplot', 'Xplot', '0', 'npexplogdens', 'labelkernel', '0', 'formatkernel', 'axtext', '6', '0.38', 'N0', 'pointsformat', 'N', 'axlegendloc', 'upper left', 'axplot', 'X', '0', '0.005', '0.01', 'np.random.random', 'Xshape', '0', 'k', 'axsetxlim', '4', '9', 'axsetylim', '0.02', '0.4', 'pltshow']:
Total running time of the script: 0 minutes 01.45 seconds
524 Neural Networks
Examples concerning the sklearn.neural_network module
Note: Click here to download the full example code
5241 Visualization of MLP weights on MNIST
Sometimes looking at the learned coefficients of a neural network can provide insight into the learning behavior. For example, if weights look unstructured, maybe some were not used at all, or if very large coefficients exist, maybe regularization was too low or the learning rate too high.
524 Neural Networks 1347

scikitlearn user guide Release 0213

This example shows how to plot some of the first layer weights in a MLPClassifier trained on the MNIST dataset. The input data consists of 28x28 pixel handwritten digits leading to 784 features in the dataset. Therefore the first layer weight matrix have the shape 784 hiddenlayersizes0. We can therefore visualize a single column of the weight matrix as a 28x28 pixel image. To make the example run faster we use very few hidden units and train only for a very short time. Training longer would result in weights with a much smoother spatial appearance.

Out

| | |
|--------------------|------------|
| Iteration 1 loss | 0.32009978 |
| Iteration 2 loss | 0.15347534 |
| Iteration 3 loss | 0.11544755 |
| Iteration 4 loss | 0.09279764 |
| Iteration 5 loss | 0.07889367 |
| Iteration 6 loss | 0.07170497 |
| Iteration 7 loss | 0.06282111 |
| Iteration 8 loss | 0.05530788 |
| Iteration 9 loss | 0.04960484 |
| Iteration 10 loss | 0.04645355 |
| Training set score | 0.986800 |
| Test set score | 0.970000 |

1348 Chapter 5 Examples

```
scikitlearn user guide Release 0213
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_openml
from sklearn.neural_network import MLPClassifier
printdoc
    Load data from https://www.openml.org/d/554
X, y = fetch_openml(mnist784, version=1, return_Xy=True)
X = X[:255]
    rescale the data use the traditional train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2,
                                                    random_state=1)
mlp = MLPClassifier(hidden_layer_sizes=(100, 100), max_iter=400, alpha=1e-4,
                    solver='sgd', verbose=10, tol=1e-4, random_state=1)
mlp.fit(X_train, y_train)
    solversgd verbose=10 tol=1e-4 random_state=1
mlp = MLPClassifier(hidden_layer_sizes=(50, 50), max_iter=10, alpha=1e-4,
                    solver='sgd', verbose=10, tol=1e-4, random_state=1)
mlp.fit(X_train, y_train)
    learningrate=1e-1
mlp.score(X_train, y_train)
print('Training set score: %f' % mlp.score(X_train, y_train))
print('Test set score: %f' % mlp.score(X_test, y_test))
fig, axes = plt.subplots(4, 4)
    use global min-max to ensure all weights are shown on the same scale
vmin, vmax = mlp.coefs_[0].min(), mlp.coefs_[0].max()
for coef, ax in zip(mlp.coefs_[0:T], axes.ravel()):
    ax.matshow(coef.reshape(28, 28), cmap=plt.cm.gray, vmin=5 * vmin,
                vmax=5 * vmax)
ax.set_xticks()
ax.set_yticks()
plt.show()
Total running time of the script: 0 minutes 27.631 seconds
Note: Click here to download the full example code
5242 Restricted Boltzmann Machine features for digit classification
For greyscale image data where pixel values can be interpreted as degrees of blackness on a white background like
handwritten digit recognition the Bernoulli Restricted Boltzmann machine model BernoulliRBM can perform
effective nonlinear feature extraction
In order to learn good latent representations from a small dataset we artificially generate more labeled data by per-
turbating the training data with linear shifts of 1 pixel in each direction
This example shows how to build a classification pipeline with a BernoulliRBM feature extractor and a
LogisticRegression classifier. The hyperparameters of the entire model (learning rate, hidden layer size, regu-
larization) were optimized by grid search but the search is not reproduced here because of runtime constraints.
Logistic regression on raw pixel values is presented for comparison. The example shows that the features extracted by
the BernoulliRBM help improve the classification accuracy.
```

scikitlearn user guide Release 0213

Out

BernoulliRBM Iteration 1 pseudolikelihood 2539 time 015s
BernoulliRBM Iteration 2 pseudolikelihood 2372 time 034s
BernoulliRBM Iteration 3 pseudolikelihood 2272 time 033s
BernoulliRBM Iteration 4 pseudolikelihood 2186 time 032s
BernoulliRBM Iteration 5 pseudolikelihood 2166 time 035s
BernoulliRBM Iteration 6 pseudolikelihood 2100 time 037s
BernoulliRBM Iteration 7 pseudolikelihood 2075 time 035s
BernoulliRBM Iteration 8 pseudolikelihood 2052 time 034s
BernoulliRBM Iteration 9 pseudolikelihood 2038 time 031s
BernoulliRBM Iteration 10 pseudolikelihood 2023 time 035s
BernoulliRBM Iteration 11 pseudolikelihood 2002 time 034s
BernoulliRBM Iteration 12 pseudolikelihood 1993 time 035s
BernoulliRBM Iteration 13 pseudolikelihood 1971 time 033s
BernoulliRBM Iteration 14 pseudolikelihood 1969 time 033s
BernoulliRBM Iteration 15 pseudolikelihood 1961 time 033s
BernoulliRBM Iteration 16 pseudolikelihood 1957 time 032s
BernoulliRBM Iteration 17 pseudolikelihood 1936 time 033s
BernoulliRBM Iteration 18 pseudolikelihood 1922 time 036s
BernoulliRBM Iteration 19 pseudolikelihood 1931 time 035s
BernoulliRBM Iteration 20 pseudolikelihood 1921 time 034s

Logistic regression using RBM features

precision recall f1score support

0 099 098 099 174

1 094 095 094 184

2 089 096 092 166

3 093 088 090 194

4 096 094 095 186

5 092 090 091 181

6 098 098 098 207

1350 Chapter 5 Examples

```
scikitlearn user guide Release 0213
7 092 099 096 154
8 087 084 086 182
9 091 091 091 169
accuracy 093 1797
macro avg 093 093 093 1797
weighted avg 093 093 093 1797
Logistic regression using raw pixel features
precision recall f1score support
0 090 091 091 174
1 060 058 059 184
2 075 085 080 166
3 078 078 078 194
4 081 084 083 186
5 076 077 077 181
6 091 087 089 207
7 085 088 087 154
8 067 057 062 182
9 075 077 076 169
accuracy 078 1797
macro avg 078 078 078 1797
weighted avg 078 078 078 1797
printdoc
  Authors Yann N Dauphin Vlad Niculae Gabriel Synnaeve
  License BSD (Binary Search Decision)
import numpy as np
import matplotlib.pyplot as plt
from scipy.ndimage import convolve
from sklearn import linear_model datasets metrics
from sklearn.model_selection import train_test_split
from sklearn.neural_network import BernoulliRBM
from sklearn.pipeline import Pipeline
from sklearn.base import clone

Setting up
def nudged_dataset(X, Y)

This produces a dataset 5 times bigger than the original one
by moving the 8x8 images in X around by 1px to left right down up

direction_vectors
0 1 0
524 Neural Networks 1351
```

scikitlearn user guide Release 0213

```
0 0 0
0 0 0
0 0 0
1 0 0
0 0 0
0 0 0
0 0 1
0 0 0
0 0 0
0 0 0
0 1 0
defshiftx w
returnconvolveXreshape8 8 modeconstant weightsw.ravel
X np.concatenateX
np.apply_along_axisshift 1 X vector
forvectorindirectionvectors
Y np.concatenateY forinrange5 axis0
returnX Y
Load Data
digits datasets.load_digits
X np.asarray(digits.data, dtype=float32)
X Y nudge_dataset(X, digit_target)
X X np.min(X, 0) np.max(X, 0) 0.0001 0.1 scaling
Xtrain Xtest Ytrain Ytest train_test_split
X Y test_size=0.2 random_state=0
Models we will use
logistic linear_model.LogisticRegressionSolverNewtonCG tol=1
multiclassmultinomial
rbm BernoulliRBM(random_state=0, verbose=True)
rbm_features_classifier Pipeline
stepsrbm rbm logistic logistic

Training
Hyperparameters These were set by crossvalidation
using a GridSearchCV Here we are not performing crossvalidation to
save time
rbm.learning_rate 0.06
rbm.niter 20
More components tend to give better prediction performance but larger
fitting time
rbm.ncomponents 100
logistic.C 6000
Training RBMLogistic Pipeline
rbm_features_classifier.fit(Xtrain, Ytrain)
1352 Chapter 5 Examples
```

scikitlearn user guide Release 0213

Training the Logistic regression classifier directly on the pixel
rawpixelclassifier clonelogistic
rawpixelclassifierC 100
rawpixelclassifierfitXtrain Ytrain

Evaluation
Ypred rbmfeaturesclassifierpredictXtest
printLogistic regression using RBM features nsn
metricsclassificationreportYtest Ypred
Ypred rawpixelclassifierpredictXtest
printLogistic regression using raw pixel features nsn
metricsclassificationreportYtest Ypred

Plotting
pltfigurefigsize42 4
fori comp inenumeratorbmcomponents
pltsubplot10 10 i 1
pltimshowcompreshape8 8 cmappltcmgrayr
interpolationnearest
pltxticks
pltyscale
pltsubplotsadjust008 002 092 085 008 023
pltshow

Total running time of the script 0 minutes 11939 seconds

Note [Click here to download the full example code](#)

5243 Varying regularization in Multilayer Perceptron

A comparison of different values for regularization parameter ‘alpha’ on synthetic datasets The plot shows that different alphas yield different decision functions

Alpha is a parameter for regularization term aka penalty term that combats overfitting by constraining the size of the weights Increasing alpha may fix high variance a sign of overfitting by encouraging smaller weights resulting in a decision boundary plot that appears with lesser curvatures Similarly decreasing alpha may fix high bias a sign of underfitting by encouraging larger weights potentially resulting in a more complicated decision boundary

524 Neural Networks 1353

```
scikitlearn user guide Release 0213
printdoc
  Author Issam H Laradji
  License BSD (Binary Search Decision) 3 clause
import numpy as np
from matplotlib import pyplotasplt
from matplotlibcolors import ListedColormap
from sklearnmodelselection import traintestsplit
from sklearnpreprocessing import StandardScaler
from sklearndatasets import makemoons makecircles makeclassification
from sklearnneuralnetwork import MLPClassifier
h 02 step size in the mesh
alphas nplogspace5 3 5
names alpha stri foriinalphas
classifiers
foriinalphas
classifiersappendMLPClassifiersolverlbfgs alphai randomstate1
hiddenlayersizes100 100
X y makeclassificationnfeatures2 nredundant0 ninformative2
randomstate0 nclustersperclass1
rng nprandomRandomState2
X 2rnguniformsizeXshape
linearlyseparable X y
datasets makemoonsnoise03 randomstate0
makecirclesnoise02 factor05 randomstate1
linearlyseparable
figure pltfigurefigsize17 9
1354 Chapter 5 Examples
```



```
i 1
    iterate over datasets
forX yindatasets
    preprocess dataset split into training and test part
X StandardScalerfittransformX
Xtrain Xtest ytrain ytest traintestsplitX y testsize4
xmin xmax X 0min 5 X 0max 5
ymin ymax X 1min 5 X 1max 5
xx yy npmeshgridnparangexmin xmax h
nparangeymax ymax h
    just plot the dataset first
cm pltcmRdBu
cmbright ListedColormapFF0000 0000FF
ax pltsubplotlendatasets lenclassifiers 1 i
    Plot the training points
axscatterXtrain 0 Xtrain 1 cytrain cmapcmbright
    and testing points
axscatterXtest 0 Xtest 1 cytest cmapcmbright alpha06
axsetxlimxxmin xmax
axsetylimyymin ymax
axsetxticks
axsetyticks
i 1
    iterate over classifiers
forname clf inzipnames classifiers
ax pltsubplotlendatasets lenclassifiers 1 i
    clffitXtrain ytrain
score clfscoreXtest ytest
    Plot the decision boundary For that we will assign a color to each
    point in the mesh xmin xmaxxymin ymax
ifhasattrclf decisionfunction
    Z clfdecisionfunctionnpcxxravel yyravel
else
    Z clfpredictprobanpcxxravel yyravel 1
    Put the result into a color plot
    Z Zreshapexxshape
    axcontourfxx yy Z cmapcm alpha8
    Plot also the training points
    axscatterXtrain 0 Xtrain 1 cytrain cmapcmbright
    edgecolorsblack s25
    and testing points
    axscatterXtest 0 Xtest 1 cytest cmapcmbright
    alpha06 edgecolorsblack s25
    axsetxlimxxmin xmax
    axsetylimyymin ymax
    axsetxticks
    axsetyticks
    axsettitlename
    axtextxxmax 3 yymin 3 2f scorelstrip0
    size15 horizontalalignmentright
i 1
524 Neural Networks 1355
```

scikitlearn user guide Release 0.21.3
fig, axes, subplots, adjustleft, 0.2, right, 0.98
plt.show()
Total running time of the script: 0 minutes 75.09 seconds
Note: Click here to download the full example code
5244 Compare Stochastic learning strategies for MLPClassifier
This example visualizes some training loss curves for different stochastic learning strategies including SGD and Adam. Because of time constraints we use several small datasets for which LBFGS might be more suitable. The general trend shown in these examples seems to carry over to larger datasets, however.
Note that those results can be highly dependent on the value of learning_rate/init.
Out:
learning on dataset iris
training constant learning_rate
Training set score: 0.980000
Training set loss: 0.096950
training constant with momentum
Training set score: 0.980000
Training set loss: 0.049530
training constant with Nesterov's momentum
1356 Chapter 5 Examples

scikitlearn user guide Release 0213
Training set score 0980000
Training set loss 0049540
training invscaling learningrate
Training set score 0360000
Training set loss 0978444
training invscaling with momentum
Training set score 0860000
Training set loss 0503452
training invscaling with Nesterovs momentum
Training set score 0860000
Training set loss 0504185
training adam
Training set score 0980000
Training set loss 0045311
learning on dataset digits
training constant learningrate
Training set score 0956038
Training set loss 0243802
training constant with momentum
Training set score 0992766
Training set loss 0041297
training constant with Nesterovs momentum
Training set score 0993879
Training set loss 0042898
training invscaling learningrate
Training set score 0638843
Training set loss 1855465
training invscaling with momentum
Training set score 0912632
Training set loss 0290584
training invscaling with Nesterovs momentum
Training set score 0909293
Training set loss 0318387
training adam
Training set score 0991653
Training set loss 0045934
learning on dataset circles
training constant learningrate
Training set score 0840000
Training set loss 0601052
training constant with momentum
Training set score 0940000
Training set loss 0157334
training constant with Nesterovs momentum
Training set score 0940000
Training set loss 0154453
training invscaling learningrate
Training set score 0500000
Training set loss 0692470
training invscaling with momentum
Training set score 0500000
Training set loss 0689143
training invscaling with Nesterovs momentum
Training set score 0500000
Training set loss 0689751
training adam
524 Neural Networks 1357

scikitlearn user guide Release 0213
Training set score 0940000
Training set loss 0150527
learning on dataset moons
training constant learningrate
Training set score 0850000
Training set loss 0341523
training constant with momentum
Training set score 0850000
Training set loss 0336188
training constant with Nesterovs momentum
Training set score 0850000
Training set loss 0335919
training invscaling learningrate
Training set score 0500000
Training set loss 0689015
training invscaling with momentum
Training set score 0830000
Training set loss 0512595
training invscaling with Nesterovs momentum
Training set score 0830000
Training set loss 0513034
training adam
Training set score 0930000
Training set loss 0170087
printdoc
import warnings
import matplotlib.pyplot as plt
from sklearnneuralnetwork import MLPClassifier
from sklearnpreprocessing import MinMaxScaler
from sklearn import datasets
from sklearnexceptions import ConvergenceWarning
different learning rate schedules and momentum parameters
params solver sgd learningrate constant momentum 0
learningrateinit 02
solver sgd learningrate constant momentum 9
nesterovsmomentum False learningrateinit 02
solver sgd learningrate constant momentum 9
nesterovsmomentum True learningrateinit 02
solver sgd learningrate invscaling momentum 0
learningrateinit 02
solver sgd learningrate invscaling momentum 9
nesterovsmomentum True learningrateinit 02
solver sgd learningrate invscaling momentum 9
nesterovsmomentum False learningrateinit 02
solver adam learningrateinit 001
labels constant learningrate constant with momentum
1358 Chapter 5 Examples

```
scikitlearn user guide Release 0213
constant with Nesterovs momentum
invscaling learningrate invscaling with momentum
invscaling with Nesterovs momentum adam
plotargs c red linestyle
c green linestyle
c blue linestyle
c red linestyle
c green linestyle
c blue linestyle
c black linestyle
defplotondatasetX y ax name
    for each dataset plot learning for each learning strategy
printlearning on dataset s name
axsettitlename
X MinMaxScalerfittransformX
mlps
ifname digits
    digits is larger but converges fairly quickly
maxiter 15
else
    maxiter 400
forlabel param inziplabels params
printtraining s label
mlp MLPClassifiervbose0 randomstate0
maxitermaxiter param
    some parameter combinations will not converge as can be seen on the
    plots so they are ignored here
withwarningscatchwarnings
warningsfilterwarningsignore categoryConvergenceWarning
modulesklearn
mlpfitX y
mlpsappendmlp
printTraining set score f mlpscoreX y
printTraining set loss f mlploss
formlp label args inzipmlps labels plotargs
axplotmlplosscurve labellabel args
fig axes pltsubplots2 2 figsize15 10
    load generate some toy datasets
iris datasetsloadiris
digits datasetsloaddigits
datasets irisdata iristarget
digitsdata digitstarget
datasetsmakecirclesnoise02 factor05 randomstate1
datasetsmakemoonsnoise03 randomstate0
forax data name inzipaxesravel datasets iris digits
circles moons
plotondataset data axax namename
524 Neural Networks 1359
```

scikitlearn user guide Release 0213
figlegendaxgetlines labels ncol3 locupper center
pltshow
Total running time of the script 0 minutes 4379 seconds

525 Preprocessing

Examples concerning the sklearnpreprocessing module

Note Click here to download the full example code

5251 Using FunctionTransformer to select columns

Shows how to use a function transformer in a pipeline If you know your dataset’s first principle component is irrelevant

for a classification task you can use the FunctionTransformer to select all but the first column of the PCA (Principal Component Analysis) transform
data

-

1360 Chapter 5 Examples

scikitlearn user guide Release 0213

```
•
import matplotlib.pyplot as plt
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA (Principal Component Analysis)
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import FunctionTransformer
def generate_vector(shift=0.5, noise=15):
    return np.arange(1000) * np.random.rand(1000) * shift + noise
def generate_dataset:
```

This dataset is two lines with a slope 1 where one has a y offset of 100

```
return np.vstack(
    np.vstack(
        generate_vector
        generate_vector * 100
    )
    np.vstack(
        generate_vector
        generate_vector
    )
)
525 Preprocessing 1361
```

```
scikitlearn user guide Release 0213
nphstacknpzeros1000 npones1000
defallbutfirstcolumnX
returnX 1
defdropfirstcomponentX y
```

Create a pipeline with PCA (Principal Component Analysis) and the column selector and use it to transform the dataset

```
pipeline makepipeline
PCA (Principal Component Analysis) FunctionTransformerallbutfirstcolumn
```

```
Xtrain Xtest ytrain ytest traintestsplitX y
pipelinefitXtrain ytrain
returnpipelinetransformXtest ytest
ifname main
X y generatedataset
lw 0
pltfigure
pltscatterX 0 X 1 cy lwlw
pltfigure
Xtransformed ytransformed dropfirstcomponent generatedataset
pltscatter
Xtransformed 0
npzeroslenXtransformed
cytransformed
lwlw
s60
```

pltshow
Total running time of the script 0 minutes 0028 seconds
Note Click here to download the full example code
5252 Using KBinsDiscretizer to discretize continuous features
The example compares prediction result of linear regression linear model and decision tree tree based model with and without discretization of realvalued features
As is shown in the result before discretization linear model is fast to build and relatively straightforward to interpret but can only model linear relationships while decision tree can build a much more complex model of the data One way to make linear model more powerful on continuous data is to use discretization also known as binning In the example we discretize the feature and onehot encode the transformed data Note that if the bins are not reasonably wide there would appear to be a substantially increased risk of overfitting so the discretizer parameters should usually be tuned under cross validation
After discretization linear regression and decision tree make exactly the same prediction As features are constant within each bin any model must predict the same value for all points within a bin Compared with the result before
1362 Chapter 5 Examples

scikitlearn user guide Release 0213

discretization linear model become much more flexible while decision tree gets much less flexible Note that binning features generally has no beneficial effect for treebased models as these models can learn to split up the data anywhere

```
Author Andreas Müller
Hanmin Qin qinhanmin2005sinacom
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import KBinsDiscretizer
from sklearn.tree import DecisionTreeRegressor
printdoc
    construct the dataset
rnd = np.random.RandomState(42)
X = rnd.uniform(3, 3, size=100)
y = np.sin(X) + rnd.normal(size=len(X) - 3)
X = X.reshape(1, 1)
    transform the dataset with KBinsDiscretizer
enc = KBinsDiscretizer(nbins=10, encode_one_hot=True)
X_binned = enc.fit_transform(X)
    predict with original dataset
fig, ax1, ax2 = plt.subplots(ncols=2, sharey=True, figsize=(10, 4))
line = np.linspace(3, 3, 1000, endpoint=False).reshape(1, 1)
reg = LinearRegression().fit(X, y)
ax1.plot(line, reg.predict(line), linewidth=2, color='green')
label('linear regression')
reg = DecisionTreeRegressor(min_samples_split=3, random_state=0).fit(X, y)
ax1.plot(line, reg.predict(line), linewidth=2, color='red')
label('decision tree')
ax1.plot(X, y, 'o', color='black')
ax1.legend(loc='best')
ax1.set_ylabel('Regression output')
ax1.set_xlabel('Input feature')
ax1.set_title('Result before discretization')
525 Preprocessing 1363
```

scikitlearn user guide Release 0213

```
predict with transformed dataset
linebinned enctransformline
reg LinearRegressionfitXbinned y
ax2plotline regpredictlinebinned linewidth2 colorgreen
linestyle labellinear regression
reg DecisionTreeRegressorminsamplesplit3
randomstate0fitXbinned y
ax2plotline regpredictlinebinned linewidth2 colorred
linestyle labeldecision tree
ax2plotX 0 y o ck
ax2vlinesencbinedges0 pltgcagetylim linewidth1 alpha2
ax2legendlocbest
ax2setxlabelInput feature
ax2settitleResult after discretization
plttightlayout
pltshow
```

Total running time of the script 0 minutes 0122 seconds

Note [Click here to download the full example code](#)

5253 Demonstrating the different strategies of KBinsDiscretizer

This example presents the different strategies implemented in KBinsDiscretizer

- ‘uniform’ The discretization is uniform in each feature which means that the bin widths are constant in each dimension
- ‘quantile’ The discretization is done on the quantiled values which means that each bin has approximately the same number of samples
- ‘kmeans’ The discretization is based on the centroids of a KMeans clustering procedure

The plot shows the regions where the discretized encoding is constant

1364 Chapter 5 Examples

```
scikitlearn user guide Release 0213
Author Tom Dupré la Tour
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import KBinsDiscretizer
from sklearn.datasets import makeblobs
printdoc
strategies uniform quantile kmeans
nsamples 200
centers0 nparray0 0 0 5 2 4 8 8
centers1 nparray0 0 3 1
construct the datasets
randomstate 42
Xlist
nprandomRandomStaterandomstateuniform3 3 sizensamples 2
makeblobsnsamplesnsamples 10 nsamples 4 10
nsamples 10 nsamples 4 10
clusterstd05 centerscenters0
randomstaterandomstate0
makeblobsnsamplesnsamples 5 nsamples 4 5
clusterstd05 centerscenters1
randomstaterandomstate0
```

```
scikitlearn user guide Release 0213
figure pltfigurefigsize14 9
i 1
for ds cnt X in enumerate(Xlist)
ax pltsubplotlenXlist lenstrategies 1 i
axscatterX 0 X 1 edgecolorsk
if ds cnt 0
axsettitleInput data size14
xx yy npmeshgrid
nplinspaceX 0min X 0max 300
nplinspaceX 1min X 1max 300
grid npcxxravel yy.ravel
axsetxlimxxmin xxmax
axsetylimyymin yymin
axsetxticks
axsetyticks
i 1
transform the dataset with KBinsDiscretizer
for strategy in strategies
enc KBinsDiscretizer nbins4 encodeordinal strategy strategy
enc fitX
grid encoded enc transform grid
ax pltsubplotlenXlist lenstrategies 1 i
horizontal stripes
horizontal grid encoded 0 reshape xxshape
axcontourfxx yy horizontal alpha5
vertical stripes
vertical grid encoded 1 reshape xxshape
axcontourfxx yy vertical alpha5
axscatterX 0 X 1 edgecolorsk
axsetxlimxxmin xxmax
axsetylimyymin yymin
axsetxticks
axsetyticks
if ds cnt 0
axsettitlestrategy s strategy size14
i 1
plt.tight_layout
plt.show
Total running time of the script 0 minutes 0890 seconds
Note Click here to download the full example code
1366 Chapter 5 Examples
```

scikitlearn user guide Release 0213

5254 Importance of Feature Scaling

Feature scaling through standardization or Zscore normalization can be an important preprocessing step for many machine learning algorithms Standardization involves rescaling the features such that they have the properties of a standard normal distribution with a mean of zero and a standard deviation of one

While many algorithms such as SVM (Support Vector Machine) Knearest neighbors and logistic regression require features to be normalized intuitively we can think of Principle Component Analysis PCA (Principal Component Analysis) as being a prime example of when normalization is important In PCA (Principal Component Analysis) we are interested in the components that maximize the variance If one component eg human height varies less than another eg weight because of their respective scales meters vs kilos PCA (Principal Component Analysis) might determine that the direction of maximal variance more closely corresponds with the 'weight' axis if those features are not scaled As a change in height of one meter can be considered much more important than the change in weight of one kilogram this is clearly incorrect

To illustrate this PCA (Principal Component Analysis) is performed comparing the use of data with StandardScaler applied to unscaled data The results are visualized and a clear difference noted The 1st principal component in the unscaled set can be seen It can be seen that feature 13 dominates the direction being a whole two orders of magnitude above the other features This is contrasted when observing the principal component for the scaled version of the data In the scaled version the orders of magnitude are roughly the same across all the features

The dataset used is the Wine Dataset available at UCI This dataset has continuous features that are heterogeneous in scale due to differing properties that they measure ie alcohol content and malic acid

The transformed data is then used to train a naive Bayes classifier and a clear difference in prediction accuracies is observed wherein the dataset which is scaled before PCA (Principal Component Analysis) vastly outperforms the unscaled version

Out

```
scikitlearn user guide Release 0213
Prediction accuracy for the normal test dataset with PCA (Principal Component Analysis)
8148
Prediction accuracy for the standardized test dataset with PCA (Principal Component Analysis)
9815
PC 1 without scaling
 176e03 836e04 155e04 531e03 202e02 102e03 153e03
112e04 631e04 233e03 154e04 743e04 100e00
PC 1 with scaling
 013 026 001 023 016 039 042 028 033 011 03 038
028
from sklearnmodelselection import traintestsplit
from sklearnpreprocessing import StandardScaler
from sklearndecomposition import PCA (Principal Component Analysis)
from sklearnnaivebayes import GaussianNB
from sklearn import metrics
import matplotlib.pyplot as plt
from sklearndatasets import loadwine
from sklearnpipeline import makepipeline
printdoc
Code source Tyler Lanigan tylerlanigangmailcom
Sebastian Raschka mailsebastianraschkacom
License BSD (Binary Search Decision) 3 clause
RANDOMSTATE 42
FIGSIZE 10 7
features target loadwinereturnXyTrue
Make a traintest split using 30 test size
Xtrain Xtest ytrain ytest traintestsplitfeatures target
testsize030
randomstateRANDOMSTATE
Fit to data and predict using pipelined GNB and PCA (Principal Component Analysis)
unscaledclf makepipelinePCAncomponents2 GaussianNB
unscaledclffitXtrain ytrain
predtest unscaledclfpredictXtest
Fit to data and predict using pipelined scaling GNB and PCA (Principal Component Analysis)
stdclf makepipelineStandardScaler PCAncomponents2 GaussianNB
stdclffitXtrain ytrain
predteststd stdclfpredictXtest
Show prediction accuracies in scaled and unscaled data
1368 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printnPrediction accuracy for the normal test dataset with PCA (Principal Component Analysis)
print2 nformatmetricsaccuracyscoreytest predtest
printnPrediction accuracy for the standardized test dataset with PCA (Principal Component Analysis)
print2 nformatmetricsaccuracyscoreytest predteststd
  Extract PCA (Principal Component Analysis) from pipeline
pca  unscaledclfnamedstepspca
pcastd  stdclfnamedstepspca
  Show first principal components
printnPC 1 without scaling n pcacomponents0
printnPC 1 with scaling n pcastdcomponents0
  Use PCA (Principal Component Analysis) without and with scale on Xtrain data for visualization
Xtraintransformed  pcatransformXtrain
scaler  stdclfnamedstepsstandardscaler
Xtrainstdtransformed  pcastdtransformscalertransformXtrain
  visualize standardized vs untouched dataset with PCA (Principal Component Analysis) performed
fig ax1 ax2  pltsubplotsncols2 figsizeFIGSIZE
forl c m inziprange0 3 blue red green  s o
ax1scatterXtraintransformedytrain l 0
Xtraintransformedytrain l 1
colorc
labelclass s l
alpha05
markerm

forl c m inziprange0 3 blue red green  s o
ax2scatterXtrainstdtransformedytrain l 0
Xtrainstdtransformedytrain l 1
colorc
labelclass s l
alpha05
markerm

ax1settitleTraining dataset after PCA (Principal Component Analysis)
ax2settitleStandardized training dataset after PCA (Principal Component Analysis)
foraxinax1 ax2
axsetxlabel1st principal component
axsetylabel2nd principal component
axlegendlocupper right
axgrid
plttightlayout
pltshow
Total running time of the script  0 minutes 0109 seconds
525 Preprocessing 1369
```

scikitlearn user guide Release 0213

Note [Click here to download the full example code](#)

5255 Map data to a normal distribution

This example demonstrates the use of the BoxCox and YeoJohnson transforms through preprocessing

PowerTransformer to map data from various distributions to a normal distribution

The power transform is useful as a transformation in modeling problems where homoscedasticity and normality are de

sired Below are examples of BoxCox and YeoJohnwon applied to six different probability distributions Lognormal

Chisquared Weibull Gaussian Uniform and Bimodal

Note that the transformations successfully map the data to a normal distribution when applied to certain datasets but

are ineffective with others This highlights the importance of visualizing the data before and after transformation

Also note that even though BoxCox seems to perform better than YeoJohnson for lognormal and chisquared distri

butions keep in mind that BoxCox does not support inputs with negative values

For comparison we also add the output from preprocessingQuantileTransformer It can force any ar

bitrary distribution into a gaussian provided that there are enough training samples thousands Because it is a

nonparametric method it is harder to interpret than the parametric ones BoxCox and YeoJohnson

On “small” datasets less than a few hundred points the quantile transformer is prone to overfitting The use of the

power transform is then recommended

1370 Chapter 5 Examples

scikitlearn user guide Release 0213
Author Eric Chang ericchang2017unorthwesternedu
Nicolas Hug contactnicolashugcom
License BSD (Binary Search Decision) 3 clause
import numpy as np
525 Preprocessing 1371

```
scikitlearn user guide Release 0213
import matplotlib.pyplot as plt
from sklearn.preprocessing import PowerTransformer
from sklearn.preprocessing import QuantileTransformer
from sklearn.model_selection import train_test_split
printdoc
NSAMPLES 1000
FONTSIZE 6
BINS 30
rng = np.random.RandomState(304)
bc = PowerTransformer(method='boxcox')
yj = PowerTransformer(method='yeo-johnson')
nquantiles is set to the training set size rather than the default value
to avoid a warning being raised by this example
qt = QuantileTransformer(nquantiles=500, output_distribution='normal',
random_state=rng,
size = NSAMPLES)
lognormal distribution
Xlognormal = rng.lognormal(size=size)
chisquared distribution
df = 3
Xchisq = rng.chisquare(df=df, size=size)
weibull distribution
a = 50
Xweibull = rng.weibullaa(size=size)
gaussian distribution
loc = 100
Xgaussian = rng.normal(loc=loc, size=size)
uniform distribution
Xuniform = rng.uniform(low=0, high=1, size=size)
bimodal distribution
loca, locb = 100, 105
Xa, Xb = rng.normal(loc=loca, size=size), rng.normal(loc=locb, size=size)
Xbimodal = np.concatenate(Xa, Xb, axis=0)
create plots
distributions
Lognormal Xlognormal
Chisquared Xchisq
Weibull Xweibull
Gaussian Xgaussian
Uniform Xuniform
Bimodal Xbimodal
```

```
scikitlearn user guide Release 0213
colors  firebrick darkorange goldenrod
seagreen royalblue darkorchid
fig axes  pltsubplotsnrows8 ncols3 figsizepltfigaspect2
axes  axesflatten
axesidxs  0 3 6 9 1 4 7 10 2 5 8 11 12 15 18 21
13 16 19 22 14 17 20 23
axeslist  axesi axesj axesk axesl
for i j k l in axesidxs
fordistribution color axes inzipdistributions colors axeslist
name X  distribution
Xtrain Xtest  traintestsplitted testsize5
perform power transforms and quantile transform
Xtransbc  bcfittedXtraintransformXtest
lmbdabc  roundbclambdas0 2
Xtransyj  yjfitXtraintransformXtest
lmbdayj  roundyjlamdas0 2
Xtransqt  qtfitXtraintransformXtest
axoriginal axbc axyj axqt  axes
axoriginalhistXtrain colorcolor binsBINS
axoriginalsettitle name fontsizeFONTSIZE
axoriginaltickparamsaxisboth whichmajor labelsizeFONTSIZE
for ax Xtrans methname lmbda in zip
axbc axyj axqt
Xtransbc Xtransyj Xtransqt
BoxCox YeoJohnson Quantile transform
lmbdabc lmbdayj None
axhistXtrans colorcolor binsBINS
title  After formatmethname
if lmbda is not None
title  rnlmbda formatlmbda
axsettitle title fontsizeFONTSIZE
axtickparamsaxisboth whichmajor labelsizeFONTSIZE
axsetxlim35 35
plt.tight_layout
plt.show
Total running time of the script 0 minutes 1536 seconds
Note Click here to download the full example code
5256 Feature discretization
A demonstration of feature discretization on synthetic classification datasets Feature discretization decomposes each
feature into a set of bins here equally distributed in width The discrete values are then onehot encoded and given to
a linear classifier This preprocessing enables a nonlinear behavior even though the classifier is linear
525 Preprocessing 1373
```

scikitlearn user guide Release 0213

On this example the first two rows represent linearly nonseparable datasets moons and concentric circles while the third is approximately linearly separable On the two linearly nonseparable datasets feature discretization largely increases the performance of linear classifiers On the linearly separable dataset feature discretization decreases the performance of linear classifiers Two nonlinear classifiers are also shown for comparison This example should be taken with a grain of salt as the intuition conveyed does not necessarily carry over to real datasets Particularly in highdimensional spaces data can more easily be separated linearly Moreover using feature discretization and onehot encoding increases the number of features which easily lead to overfitting when the number of samples is small The plots show training points in solid colors and testing points semitransparent The lower right shows the classification accuracy on the test set

Out

dataset 0

LogisticRegression 086
LinearSVC 086
KBinsDiscretizer LogisticRegression 094
KBinsDiscretizer LinearSVC 092
GradientBoostingClassifier 090
SVC (Support Vector Classification) 094
dataset 1

LogisticRegression 040
LinearSVC 040
KBinsDiscretizer LogisticRegression 088
KBinsDiscretizer LinearSVC 086
GradientBoostingClassifier 080
SVC (Support Vector Classification) 084
dataset 2

LogisticRegression 096
LinearSVC 098
KBinsDiscretizer LogisticRegression 094
KBinsDiscretizer LinearSVC 084
GradientBoostingClassifier 094
1374 Chapter 5 Examples

scikitlearn user guide Release 0213
SVC (Support Vector Classification) 098
Code source Tom Dupré la Tour
Adapted from plotclassifiercomparison by Gaël Varoquaux and Andreas Müller

```
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import make_moons, make_circles, make_classification
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import KBinsDiscretizer
from sklearn.svm import SVC (Support Vector Classification) LinearSVC
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.utils.testing import ignore_warnings
from sklearn.exceptions import ConvergenceWarning
printdoc
h_02_step_size_in_the_mesh
def get_name_estimator
name_estimator_classname
if name_pipeline
name_get_name_est1_forest_in_estimator_steps
name_joinname
return name
list of estimator paramgrid where paramgrid is used in GridSearchCV
classifiers
LogisticRegression solver lbfgs randomstate 0
C nlogspace 2 7 10

LinearSVC randomstate 0
C nlogspace 2 7 10

make_pipeline
KBinsDiscretizer encode_one_hot
LogisticRegression solver lbfgs randomstate 0
kbin_discretizer nbins nparange 2 10
logistic_regression C nlogspace 2 7 10

make_pipeline
KBinsDiscretizer encode_one_hot LinearSVC randomstate 0
kbin_discretizer nbins nparange 2 10
linear_svc C nlogspace 2 7 10
525 Preprocessing 1375
```

GradientBoostingClassifiernestimators50 randomstate0
learningrate nplogspace4 0 10

SVCrandomstate0 gammascale
C nplogspace2 7 10

names_ getnamee fore ginclassifiers
nsamples_ 100
datasets
makemoonsnsamplesnsamples noise02 randomstate0
makecirclesnsamplesnsamples noise02 factor05 randomstate1
makeclassificationnsamplesnsamples nfeatures2 nredundant0
ninformative2 randomstate2
nclustersperclass1

fig axes_ pltsubplotsnrowslendatasets ncolslenclassifiers_ 1
figsize21 9
cm_ pltcmPiYG
cmbright_ ListedColormapb30065 178000
iterate over datasets
fordscnt X y inenumeratedatasets
printndataset dn_ dscnt
preprocess dataset split into training and test part
X_ StandardScalerfittransformX
Xtrain Xtest ytrain ytest_ traintestsplit
X y testsize5 randomstate42
create the grid for background colors
xmin xmax_ X 0min_ 5 X 0max_ 5
ymin ymax_ X 1min_ 5 X 1max_ 5
xx yy_ npmeshgrid
nparangexmin xmax h nparangeymax ymax h
plot the dataset first
ax_ axesdscnt 0
ifdscnt_ 0
axsettitleInput data
plot the training points
axscatterXtrain 0 Xtrain 1 cytrain cmapcmbright
edgecolorsk
and testing points
axscatterXtest 0 Xtest 1 cytest cmapcmbright alpha06
edgecolorsk
axsetxlimxxmin xxmax
axsetylimyymin yymax
axsetxticks
axsetyticks
iterate over classifiers
forestidx name estimator paramgrid in
1376 Chapter 5 Examples

```
scikitlearn user guide Release 0213
enumeratezipnames classifiers
ax = axesdscnt estidx = 1
clf = GridSearchCV(estimator=estimator, param_grid=param_grid, cv=5,
iid=False,
with_ignore_warnings=True, category=ConvergenceWarning)
clf.fit(X_train, y_train)
score = clf.score(X_test, y_test)
print('2f name score')
plot the decision boundary For that we will assign a color to each
point in the mesh
xmin, xmax, ymin, ymax
if hasattr(clf, 'decision_function'):
Z = clf.decision_function(np.c_[x.ravel(), y.ravel()])
else:
Z = clf.predict_proba(np.c_[x.ravel(), y.ravel()])
put the result into a color plot
Z = Z.reshape(x.shape)
ax.contourf(x, y, Z, cmap=cm_alpha)
plot the training points
ax.scatter(X_train[0], X_train[1], c=y_train, cmap=cm_bright)
edgecolors = k
and testing points
ax.scatter(X_test[0], X_test[1], c=y_test, cmap=cm_bright)
edgecolors = k, alpha=0.6
ax.set_xlim(xmin, xmax)
ax.set_ylim(ymin, ymax)
ax.set_xticks
ax.set_yticks
if dscnt == 0:
ax.set_title(name.replace(' ', '\n'))
ax.text(0.95, 0.06, '2f score', strip=0, size=15)
bbox=dict(boxstyle='round', alpha=0.8, facecolor='white')
transform = maxtransAxes, horizontalalignment='right'
plt.tight_layout()
Add subtitles above the figure
plt.subplots_adjust(top=0.9)
subtitles
Linear classifiers
Feature discretization and linear classifiers
Nonlinear classifiers

for i, subtitle in zip(1, 3, 5, subtitles):
ax = axes[0, i]
ax.text(1.05, 1.25, subtitle, transform=maxtransAxes,
horizontalalignment='center', size=large)
plt.show()
Total running time of the script: 0 minutes 16.451 seconds
525 Preprocessing 13.77
```

scikitlearn user guide Release 0213

Note Click here to download the full example code

5257 Compare the effect of different scalers on data with outliers

Feature 0 median income in a block and feature 5 number of households of the California housing dataset have very different scales and contain some very large outliers These two characteristics lead to difficulties to visualize the data and more importantly they can degrade the predictive performance of many machine learning algorithms Unscaled data can also slow down or even prevent the convergence of many gradientbased estimators Indeed many estimators are designed with the assumption that each feature takes values close to zero or more importantly that all features vary on comparable scales In particular metricbased and gradientbased estimators often assume approximately standardized data centered features with unit variances A notable exception are decision treebased estimators that are robust to arbitrary scaling of the data

This example uses different scalers transformers and normalizers to bring the data within a predefined range Scalers are linear or more precisely affine transformers and differ from each other in the way to estimate the parameters used to shift and scale each feature

QuantileTransformer provides nonlinear transformations in which distances between marginal outliers and inliers are shrunk PowerTransformer provides nonlinear transformations in which data is mapped to a normal distribution to stabilize variance and minimize skewness

Unlike the previous transformations normalization refers to a per sample transformation instead of a per feature transformation

The following code is a bit verbose feel free to jump directly to the analysis of the results

```
Author Raghav RV rvraghav93gmailcom
Guillaume Lemaitre glemlaire58gmailcom
Thomas Unterthiner
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib as mpl
from matplotlib import pyplotasplt
from matplotlib import cm
from sklearnpreprocessing import MinMaxScaler
from sklearnpreprocessing import minmaxscale
from sklearnpreprocessing import MaxAbsScaler
from sklearnpreprocessing import StandardScaler
from sklearnpreprocessing import RobustScaler
from sklearnpreprocessing import Normalizer
from sklearnpreprocessing import QuantileTransformer
from sklearnpreprocessing import PowerTransformer
from sklearnndatasets import fetchcaliforniahousing
printdoc
dataset fetchcaliforniahousing
Xfull yfull datasetdata datasettarget
Take only 2 features to make visualization easier
1378 Chapter 5 Examples
```


scikitlearn user guide Release 0213
Feature of 0 has a long tail distribution
Feature 5 has a few but very large outliers
X Xfull 0 5
distributions
Unscaled data X
Data after standard scaling
StandardScalerfittransformX
Data after minmax scaling
MinMaxScalerfittransformX
Data after maxabs scaling
MaxAbsScalerfittransformX
Data after robust scaling
RobustScalerquantilerange25 75fittransformX
Data after power transformation YeoJohnson
PowerTransformermethodyeojohnsonfittransformX
Data after power transformation BoxCox
PowerTransformermethodboxcoxfittransformX
Data after quantile transformation gaussian pdf
QuantileTransformeroutputdistributionnormal
fittransformX
Data after quantile transformation uniform pdf
QuantileTransformeroutputdistributionuniform
fittransformX
Data after samplewise L2 normalizing
NormalizerfittransformX

scale the output between 0 and 1 for the colorbar
y minmaxscaleyfull
plasma does not exist in matplotlib 15
cmap getattrcm plasmar cmhotr
defcreateaxestitle figsize16 6
fig pltfigurefigsizefigsize
figsuptitletitle
define the axis for the first plot
left width 01 022
bottom height 01 07
bottomh height 015
lefth left width 002
rectscatter left bottom width height
recthistx left bottomh width 01
recthisty lefth bottom 005 height
axscatter pltaxesrectscatter
axhistx pltaxesrecthistx
axhisty pltaxesrecthisty
define the axis for the zoomedin plot
left width left 02
lefth left width 002
rectscatter left bottom width height
525 Preprocessing 1379

```
scikitlearn user guide Release 0213
recthistx left bottomh width 01
recthisty lefth bottom 005 height
axscatterzoom pltaxesrectscatter
axhistxzoom pltaxesrecthistx
axhistyzoom pltaxesrecthisty
    define the axis for the colorbar
left width width left 013 001
rectcolorbar left bottom width height
axcolorbar pltaxesrectcolorbar
returnaxscatter axhisty axhistx
axscatterzoom axhistyzoom axhistxzoom
axcolorbar
defplotdistributionaxes X y histnbins50 title
x0label x1label
ax histX1 histX0 axes
axsettitletitle
axsetxlabelx0label
axsetylabelx1label
    The scatter plot
colors cmapy
axscatterX 0 X 1 alpha05 markero s5 lw0 ccolors
    Removing the top and the right spine for aesthetics
    make nice axis layout
axspinesstopsetvisibleFalse
axspinesrightsetvisibleFalse
axgetxaxistickbottom
axgetyaxistickleft
axspinesleftsetpositionoutward 10
axspinesbottomsetpositionoutward 10
    Histogram for axis X1 feature 5
histX1setylimaxgetylim
histX1histX 1 binshistnbins orientationhorizontal
colorgrey ecgrey
histX1axisoff
    Histogram for axis X0 feature 0
histX0setxlimaxgetxlim
histX0histX 0 binshistnbins orientationvertical
colorgrey ecgrey
histX0axisoff
Two plots will be shown for each scalernormalizertransformer The left figure will show a scatter plot of the full data
set while the right figure will exclude the extreme values considering only 99 of the data set excluding marginal
outliers In addition the marginal distributions for each feature will be shown on the side of the scatter plot
defmakeplotitemidx
title X distributionsitemidx
axzoomout axzoomin axcolorbar createaxestitle
1380 Chapter 5 Examples
```

scikitlearn user guide Release 0213

axarr axzoomout axzoomin
plotdistributionaxarr0 X y histnbins200

x0labelMedian Income

x1labelNumber of households

titleFull data

zoomin

zoominpercentilerange 0 99

cutoffsX0 nppercentileX 0 zoominpercentilerange

cutoffsX1 nppercentileX 1 zoominpercentilerange

nonoutliersmask

npallX cutoffsX00 cutoffsX10 axis1

npallX cutoffsX01 cutoffsX11 axis1

plotdistributionaxarr1 Xnonoutliersmask ynonoutliersmask

histnbins50

x0labelMedian Income

x1labelNumber of households

titleZoomin

norm mplcolorsNormalizeyfullmin yfullmax

mplcolorbarColorbarBaseaxcolorbar cmapcmap

normnorm orientationvertical

labelColor mapping for values of y

Original data

Each transformation is plotted showing two transformed features with the left plot showing the entire dataset and the right zoomedin to show the dataset without the marginal outliers A large majority of the samples are compacted to a specific range 0 10 for the median income and 0 6 for the number of households Note that there are some marginal outliers some blocks have more than 1200 households Therefore a specific preprocessing can be very beneficial depending of the application In the following we present some insights and behaviors of those preprocessing methods in the presence of marginal outliers

makeplot0
525 Preprocessing 1381

StandardScaler

StandardScaler removes the mean and scales the data to unit variance. However, the outliers have an influence when computing the empirical mean and standard deviation, which shrink the range of the feature values as shown in the left figure below. Note in particular that because the outliers on each feature have different magnitudes, the spread of the transformed data on each feature is very different: most of the data lie in the $[-2, 4]$ range for the transformed median income feature, while the same data is squeezed in the smaller $[-0.2, 0.2]$ range for the transformed number of households.

StandardScaler therefore cannot guarantee balanced feature scales in the presence of outliers.

makeplot1

MinMaxScaler

MinMaxScaler rescales the data set such that all feature values are in the range $[0, 1]$ as shown in the right panel below. However, this scaling compresses all inliers in the narrow range $[0, 0.005]$ for the transformed number of households.

AsStandardScaler MinMaxScaler is very sensitive to the presence of outliers.

makeplot2

MaxAbsScaler

MaxAbsScaler differs from the previous scaler such that the absolute values are mapped in the range 0 1 On positive only data this scaler behaves similarly to MinMaxScaler and therefore also suffers from the presence of large outliers

makeplot3

RobustScaler

Unlike the previous scalers the centering and scaling statistics of this scaler are based on percentiles and are therefore not influenced by a few number of very large marginal outliers Consequently the resulting range of the transformed feature values is larger than for the previous scalers and more importantly are approximately similar for both features most of the transformed values lie in a 2 3 range as seen in the zoomedin figure Note that the outliers themselves are still present in the transformed data If a separate outlier clipping is desirable a nonlinear transformation is required see below

makeplot4

PowerTransformer

PowerTransformer applies a power transformation to each feature to make the data more Gaussianlike Cur

scikitlearn user guide Release 0213

rentlyPowerTransformer implements the YeoJohnson and BoxCox transforms The power transform finds the optimal scaling factor to stabilize variance and minimize skewness through maximum likelihood estimation By defaultPowerTransformer also applies zero mean unit variance normalization to the transformed output Note that BoxCox can only be applied to strictly positive data Income and number of households happen to be strictly positive but if negative values are present the YeoJohnson transformed is to be preferred

makeplot5

makeplot6

•

•

QuantileTransformer Gaussian output

QuantileTransformer has an additional outputdistribution parameter allowing to match a Gaussian distribution instead of a uniform distribution Note that this nonparametric transformer introduces saturation artifacts for extreme values

makeplot7

1384 Chapter 5 Examples

QuantileTransformer uniform output

QuantileTransformer applies a nonlinear transformation such that the probability density function of each feature will be mapped to a uniform distribution In this case all the data will be mapped in the range 0 1 even the outliers which cannot be distinguished anymore from the inliers

AsRobustScaler QuantileTransformer is robust to outliers in the sense that adding or removing outliers in the training set will yield approximately the same transformation on held out data But contrary to RobustScaler QuantileTransformer will also automatically collapse any outlier by setting them to the a priori defined range boundaries 0 and 1

makeplot8

Normalizer

TheNormalizer rescales the vector for each sample to have unit norm independently of the distribution of the samples It can be seen on both figures below where all samples are mapped onto the unit circle In our example the two selected features have only positive values therefore the transformed data only lie in the positive quadrant This would not be the case if some original features had a mix of positive and negative values

scikitlearn user guide Release 0213
makeplot9
pltshow
Total running time of the script 0 minutes 4599 seconds
526 Semi Supervised Classification
Examples concerning the sklearnsemisupervised module
Note [Click here to download the full example code](#)
5261 Decision boundary of label propagation versus SVM (Support Vector Machine) on the Iris dataset
Comparison for decision boundary generated on iris dataset between Label Propagation and SVM (Support Vector Machine)
This demonstrates Label Propagation learning a good boundary even with a small amount of labeled data
1386 Chapter 5 Examples


```
scikitlearn user guide Release 0213
printdoc
  Authors Clay Woolam claywoolamorg
  License BSD (Binary Search Decision)
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn import svm
from sklearn.semisupervised import labelpropagation
rng = np.random.RandomState(0)
iris = datasets.load_iris()
X = iris.data[:, 2:]
y = iris.target
# step size in the mesh
h = 0.2
y30 = np.copy(y)
y30[rng.randint(0, 30, 1)] = 1
y50 = np.copy(y)
y50[rng.randint(0, 50, 1)] = 1
# we create an instance of SVM (Support Vector Machine) and fit out data
# We do not scale our
526 Semi Supervised Classification 1387
```

scikitlearn user guide Release 0213

data since we want to plot the support vectors

ls30 labelpropagationLabelSpreadingfitX y30
y30

ls50 labelpropagationLabelSpreadingfitX y50
y50

ls100 labelpropagationLabelSpreadingfitX y y
rbfsvc svmSVCKernelrbf gamma5fitX y y

create a mesh to plot in

xmin xmax X 0min 1 X 0max 1

ymin ymax X 1min 1 X 1max 1

xx yy npmeshgridnparangexmin xmax h

nparangeymin ymax h

title for the plots

titles Label Spreading 30 data

Label Spreading 50 data

Label Spreading 100 data

SVC (Support Vector Classification) with rbf kernel

colormap 1 1 1 1 0 0 0 9 1 1 0 0 2 8 6 0

fori clf ytrain inenumeratels30 ls50 ls100 rbfsvc

Plot the decision boundary For that we will assign a color to each

point in the mesh xmin xmaxxymin ymax

pltsubplot2 2 i 1

Z clfpredictnpcxxravel yy.ravel

Put the result into a color plot

Z Zreshapexxshape

pltcontourfxx yy Z cmappltcmPaired

pltaxisoff

Plot also the training points

colors colormapy forinytrain

pltscatterX 0 X 1 ccolors edgecolorsblack

plttitletitlesi

pltstitleUnlabeled points are colored white y01

pltshow

Total running time of the script 0 minutes 0915 seconds

Note Click here to download the full example code

5262 Label Propagation learning a complex structure

Example of LabelPropagation learning a complex internal structure to demonstrate “manifold learning” The outer circle should be labeled “red” and the inner circle “blue” Because both label groups lie inside their own distinct shape we can see that the labels propagate correctly around the circle

1388 Chapter 5 Examples

```
scikitlearn user guide Release 0213
printdoc
Authors Clay Woolam claywoolamorg
Andreas Mueller amuelleraisunibonnde
License BSD (Binary Search Decision)
import numpy as np
import matplotlib.pyplot as plt
from sklearn.semi_supervised import LabelPropagation
from sklearn.datasets import make_circles
generate ring with inner box
nsamples = 200
X, y = make_circles(nsamples, nsamples, shuffle=False,
                    outer_inner=(0, 1),
                    labels=np.full(nsamples, 1),
                    labels0=outer,
                    labels1=inner)

# Learn with LabelSpreading
labelspred = LabelPropagation().fit(X, labels)
labelspred.fit(X, labels)

# Plot output labels
output_labels = labelspred.transduction_
plt.figure(figsize=(8, 5))
plt.subplot(2, 1, 1)
plt.scatter(X_labels=outer, X_labels=inner, color=navy,
            markersize=10, label=outer, labeled=True)
plt.scatter(X_labels=inner, X_labels=inner, color=coral,
            markersize=10, label=inner, labeled=True)
plt.scatter(X_labels=[0, 1], X_labels=[1, 1], color=darkorange,
            marker=unlabeled)
plt.legend(scatterpoints1, shadow=False, loc='upper right')
plt.title('Raw data: 2 classes: outer and inner')
526 Semi-Supervised Classification 1389
```

```
scikitlearn user guide Release 0213
plt.subplot(1 2 2)
outputlabelarray np.asarray(outputlabels)
outernumbers np.where(outputlabelarray == outer0)
innernumbers np.where(outputlabelarray == inner0)
plt.scatter(X[outernumbers, 0], X[outernumbers, 1], color='navy',
            markersize=10, label='outer learned')
plt.scatter(X[innernumbers, 0], X[innernumbers, 1], color='c',
            markersize=10, label='inner learned')
plt.legend(scatterpoints1, shadow=False, loc='upper right')
plt.title('Labels learned with Label Spreading KNN')
plt.subplots_adjust(left=0.07, bottom=0.07, right=0.93, top=0.92)
plt.show()
Total running time of the script: 0 minutes 00.31 seconds
Note: Click here to download the full example code
5263 Label Propagation digits: Demonstrating performance
This example demonstrates the power of semisupervised learning by training a Label Spreading model to classify
handwritten digits with sets of very few labels.
The handwritten digit dataset has 1797 total points. The model will be trained using all points but only 30 will be
labeled. Results in the form of a confusion matrix and a series of metrics over each class will be very good.
At the end, the top 10 most uncertain predictions will be shown.
```

scikitlearn user guide Release 0213

Out

Label Spreading model 40 labeled 300 unlabeled points 340 total

precision recall f1score support

0 100 100 100 27

1 082 100 090 37

2 100 086 092 28

3 100 080 089 35

4 092 100 096 24

5 074 094 083 34

6 089 096 092 25

7 094 089 091 35

8 100 068 081 31

9 081 088 084 24

accuracy 090 300

macro avg 091 090 090 300

weighted avg 091 090 090 300

Confusion matrix

27 0 0 0 0 0 0 0 0 0

0 37 0 0 0 0 0 0 0

0 1 24 0 0 0 2 1 0 0

0 0 0 28 0 5 0 1 0 1

0 0 0 0 24 0 0 0 0 0

0 0 0 0 0 32 0 0 0 2

526 Semi Supervised Classification 1391

```
scikitlearn user guide Release 0213
0 0 0 0 0 1 24 0 0 0
0 0 0 0 1 3 0 31 0 0
0 7 0 0 0 0 1 0 21 2
0 0 0 0 1 2 0 0 0 21
printdoc
Authors Clay Woolam claywoolamorg
License BSD (Binary Search Decision)
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
from sklearn import datasets
from sklearn.semisupervised import labelpropagation
from sklearn.metrics import confusionmatrix classificationreport
digits datasetsloaddigits
rng np.random.RandomState(2)
indices np.arange(digits.data.shape[0])
rng.shuffle(indices)
X digits.data[indices]
y digits.target[indices]
images digits.images[indices]
ntotal_samples len(y)
n_labeled_points 40
indices np.arange(ntotal_samples)
unlabeledset indices[n_labeled_points:]

# Shuffle everything around
y_train np.copy(y)
y_train[unlabeledset] = 1

# Learn with LabelSpreading
lp_model labelpropagation.LabelSpreading(gamma=25, max_iter=20)
lp_model.fit(X, y_train)
predicted_labels lp_model.transduction(unlabeledset)
true_labels y[unlabeledset]
cm confusionmatrix(true_labels, predicted_labels)
print('Label Spreading model d_labeled: %d, d_unlabeled: %d, points: %d, total: %d' %
      (cm[0][0], cm[0][1], cm[1][0], cm[1][1]))
n_labeled_points n_labeled_points
ntotal_samples ntotal_samples
1392 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printclassificationreporttruelabels predictedlabels
printConfusion matrix
printcm
```

```
Calculate uncertainty values for each transduced distribution
predentropies statsdistributionsentropyIpmodellabeldistributionsT
```

```
Pick the top 10 most uncertain labels
uncertaintyindex npargsortpredentropies10
```

```
Plot
f pltfigurefigsize7 5
forindex imageindex inenumerateuncertaintyindex
image imagesimageindex
sub faddsubplot2 5 index 1
subimshowimage cmappltcmgrayr
pltxticks
pltyticks
subsettitledpredict intruei
lpmodeltransductionimageindex yimageindex
fsuptitleLearning with small amount of labeled data
pltshow
Total running time of the script 0 minutes 0225 seconds
Note Click here to download the full example code
5264 Label Propagation digits active learning
Demonstrates an active learning technique to learn handwritten digits using label propagation
We start by training a label propagation model with only 10 labeled points then we select the top five most uncertain
points to label Next we train with 15 labeled points original 10 5 new ones We repeat this process four times
to have a model trained with 30 labeled examples Note you can increase this to label more than 30 by changing
maxiterations Labeling more than 30 can be useful to get a sense for the speed of convergence of this active
learning technique
A plot will appear showing the top 5 most uncertain digits for each iteration of training These may or may not contain
mistakes but we will train the next model with their true labels
526 Semi Supervised Classification 1393
```

scikitlearn user guide Release 0213

Out

Iteration 0

Label Spreading model 40 labeled 290 unlabeled 330 total

| | precision | recall | f1score | support |
|--------------|-----------|--------|---------|---------|
| 0 | 100 | 100 | 100 | 22 |
| 1 | 078 | 069 | 073 | 26 |
| 2 | 093 | 093 | 093 | 29 |
| 3 | 100 | 089 | 094 | 27 |
| 4 | 092 | 096 | 094 | 23 |
| 5 | 096 | 070 | 081 | 33 |
| 6 | 097 | 097 | 097 | 35 |
| 7 | 094 | 091 | 092 | 33 |
| 8 | 062 | 089 | 074 | 28 |
| 9 | 073 | 079 | 076 | 34 |
| accuracy | 087 | 087 | 087 | 290 |
| macro avg | 089 | 087 | 087 | 290 |
| weighted avg | 088 | 087 | 087 | 290 |

Confusion matrix

| | | | | | | | | |
|----|----|----|----|---|---|---|---|---|
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 2 | 0 | 0 | 0 | 1 | 0 | 5 |
| 0 | 0 | 27 | 0 | 0 | 0 | 0 | 0 | 2 |
| 0 | 0 | 0 | 24 | 0 | 0 | 0 | 0 | 3 |

1394 Chapter 5 Examples

scikitlearn user guide Release 0213

0 1 0 0 22 0 0 0 0
0 0 0 0 0 23 0 0 0 10
0 1 0 0 0 0 34 0 0 0
0 0 0 0 0 0 0 30 3 0
0 3 0 0 0 0 0 0 25 0
0 0 0 0 2 1 0 2 2 27

Iteration 1

Label Spreading model 45 labeled 285 unlabeled 330 total

precision recall f1score support

0 100 100 100 22
1 079 100 088 22
2 100 093 096 29
3 100 100 100 26
4 092 096 094 23
5 096 070 081 33
6 100 097 099 35
7 094 091 092 33
8 077 086 081 28
9 073 079 076 34

accuracy 090 285

macro avg 091 091 091 285

weighted avg 091 090 090 285

Confusion matrix

22 0 0 0 0 0 0 0 0 0
0 22 0 0 0 0 0 0 0 0
0 0 27 0 0 0 0 0 2 0
0 0 0 26 0 0 0 0 0 0
0 1 0 0 22 0 0 0 0 0
0 0 0 0 0 23 0 0 0 10
0 1 0 0 0 0 34 0 0 0
0 0 0 0 0 0 0 30 3 0
0 4 0 0 0 0 0 0 24 0
0 0 0 0 2 1 0 2 2 27

Iteration 2

Label Spreading model 50 labeled 280 unlabeled 330 total

precision recall f1score support

0 100 100 100 22
1 085 100 092 22
2 100 100 100 28
3 100 100 100 26
4 087 100 093 20
5 096 070 081 33
6 100 097 099 35
7 094 100 097 32
8 092 086 089 28
9 073 079 076 34

accuracy 092 280

macro avg 093 093 093 280

weighted avg 093 092 092 280

Confusion matrix

22 0 0 0 0 0 0 0 0 0
0 22 0 0 0 0 0 0 0 0

526 Semi Supervised Classification 1395

scikitlearn user guide Release 0213

0 0 28 0 0 0 0 0 0
0 0 0 26 0 0 0 0 0
0 0 0 0 20 0 0 0 0
0 0 0 0 23 0 0 0 10
0 1 0 0 0 0 34 0 0
0 0 0 0 0 0 32 0 0
0 3 0 0 1 0 0 0 24 0
0 0 0 0 2 1 0 2 2 27

Iteration 3
Label Spreading model 55 labeled 275 unlabeled 330 total

precision recall f1score support

0 100 100 100 22
1 085 100 092 22
2 100 100 100 27
3 100 100 100 26
4 087 100 093 20
5 096 087 092 31
6 100 097 099 35
7 100 100 100 31
8 092 086 089 28
9 088 085 086 33

accuracy 095 275
macro avg 095 095 095 275
weighted avg 095 095 095 275

Confusion matrix

22 0 0 0 0 0 0 0 0
0 22 0 0 0 0 0 0 0
0 0 27 0 0 0 0 0 0
0 0 0 26 0 0 0 0 0
0 0 0 0 20 0 0 0 0
0 0 0 0 27 0 0 0 4
0 1 0 0 0 0 34 0 0
0 0 0 0 0 0 31 0 0
0 3 0 0 1 0 0 0 24 0
0 0 0 0 2 1 0 0 2 28

Iteration 4
Label Spreading model 60 labeled 270 unlabeled 330 total

precision recall f1score support

0 100 100 100 22
1 096 100 098 22
2 100 096 098 27
3 096 100 098 25
4 086 100 093 19
5 096 087 092 31
6 100 097 099 35
7 100 100 100 31
8 092 096 094 25
9 088 085 086 33

accuracy 096 270
macro avg 095 096 096 270
weighted avg 096 096 096 270

Confusion matrix

1396 Chapter 5 Examples

scikitlearn user guide Release 0213

```
22 0 0 0 0 0 0 0 0 0
0 22 0 0 0 0 0 0 0 0
0 0 26 1 0 0 0 0 0 0
0 0 0 25 0 0 0 0 0 0
0 0 0 0 19 0 0 0 0 0
0 0 0 0 0 27 0 0 0 4
0 1 0 0 0 0 34 0 0 0
0 0 0 0 0 0 0 31 0 0
0 0 0 0 1 0 0 0 24 0
0 0 0 0 2 1 0 0 2 28
printdoc
Authors Clay Woolam claywoolamorg
License BSD (Binary Search Decision)
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
from sklearn import datasets
from sklearn.semisupervised import labelpropagation
from sklearn.metrics import classification_report confusion_matrix
digits = datasets.load_digits
rng = np.random.RandomState(0)
indices = np.arange(digits.data.shape[0])
rng.shuffle(indices)
X = digits.data[indices]
y = digits.target[indices]
images = digits.images[indices]
ntotal_samples = len(y)
n_labeled_points = 40
max_iterations = 5
unlabeled_indices = np.arange(ntotal_samples - n_labeled_points)
f = plt.figure
for i in range(max_iterations):
    if len(unlabeled_indices) == 0:
        print('No unlabeled items left to label')
        break
    y_train = y[indices]
    y_train[unlabeled_indices] = 1
    lp_model = labelpropagation.LabelSpreading(gamma=0.25, max_iter=20)
    lp_model.fit(X, y_train)
    predicted_labels = lp_model.transduction[unlabeled_indices]
    true_labels = y[unlabeled_indices]
```

```
scikitlearn user guide Release 0213
cm confusionmatrixtruelabels predictedlabels
labelsipmodelclasses
printIteration i s i 70
printLabel Spreading model dlabelled dunlabeled dtotal
nlabelledpoints ntotalsamples nlabelledpoints
ntotalsamples
printclassificationreporttruelabels predictedlabels
printConfusion matrix
printcm
compute the entropies of transduced label distributions
predentropies statsdistributionsentropy
lpmodellabeldistributionsT
select up to 5 digit examples that the classifier is most uncertain about
uncertaintyindex npargsortpredentropies1
uncertaintyindex uncertaintyindex
npin1duncertaintyindex unlabeledindices5
keep track of indices that we get labels for
deleteindices nparray dtypeint
for more than 5 iterations visualize the gain only on the first 5
ifi 5
ftext05 1 i 1 183
modeldnnfit with ndlabels
i 1 i 5 10 size10
forindex imageindex inenumerateuncertaintyindex
image imagesimageindex
for more than 5 iterations visualize the gain only on the first 5
ifi 5
sub faddsubplot5 5 index 1 5 i
subimshowimage cmappltcmgrayr interpolationnone
subsettitledpredict intruei
lpmodeltransductionimageindex yimageindex size10
subaxisoff
labeling 5 points remote from labeled set
deleteindex npwhereunlabeledindices imageindex
deleteindices npconcatenatedeleteindices deleteindex
unlabeledindices npdeleteunlabeledindices deleteindices
nlabelledpoints lenuncertaintyindex
fsuptitleActive learning with Label Propagation nRows show 5 most
uncertain labels to learn with the next model y115
pltsubplotsadjleft02 bottom003 right09 top09 wspace02
hspace085
pltshow
Total running time of the script 0 minutes 0575 seconds
1398 Chapter 5 Examples
```

scikitlearn user guide Release 0213

527 Support Vector Machines

Examples concerning the sklearnsvm module

Note Click here to download the full example code

5271 Nonlinear SVM (Support Vector Machine)

Perform binary classification using nonlinear SVC (Support Vector Classification) with RBF (Radial Basis Function) kernel The target to predict is a X

The color map illustrates the decision function learned by the SVC (Support Vector Classification)

```
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
xx,yy = np.meshgrid(np.linspace(3,3,500)
np.linspace(3,3,500)
np.random.seed(0)
X = np.random.randn(300,2)
Y = np.logical_xor(X[:,0] > 0,X[:,1] > 0)
527 Support Vector Machines 1399
```

```
scikitlearn user guide Release 0213
fit the model
clf = svm.NuSVC(gamma=auto)
clf.fit(X, Y)
plot the decision function for each datapoint on the grid
Z = clf.decision_function(np.cxx.ravel(yravel)
Z = Z.reshape(xxshape)
plt.imshow(Z, interpolation='nearest',
           extent=(xxmin, xxmax, yymin, ymax), aspect='auto',
           origin='lower', cmap=plt.cm.PuOr,
           contours=1, plt.contourxx, yy, Z, levels=0, linewidths=2,
           linestyle='dashed',
           plt.scatter(X, 0, X, 1, s=30, c=Y, cmap=plt.cm.Paired,
                      edgecolors='k',
                      plt.xticks,
                      plt.yticks,
                      plt.axis(3, 3, 3, 3),
                      plt.show)
Total running time of the script: 0 minutes 1073 seconds
Note: Click here to download the full example code
5272 SVM (Support Vector Machine) Maximum margin separating hyperplane
Plot the maximum margin separating hyperplane within a two-class separable dataset using a Support Vector Machine
classifier with linear kernel
1400 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn.datasets import makeblobs
    we create 40 separable points
X, y = makeblobs(nsamples=40, centers=2, randomstate=6)
fit the model dont regularize for illustration purposes
clf = svm.SVC(kernel='linear', C=1000)
clf.fit(X, y)
plt.scatter(X[:, 0], X[:, 1], s=30, cmap=plt.cm.Paired)
plot the decision function
ax = plt.gca()
xlim = ax.get_xlim()
ylim = ax.get_ylim()
    create grid to evaluate model
xx = np.linspace(xlim[0], xlim[1], 30)
yy = np.linspace(ylim[0], ylim[1], 30)
YY, XX = np.meshgrid(yy, xx)
527 Support Vector Machines 1401
```

```
scikitlearn user guide Release 0213
xy npvstackXXravel YYravelT
Z clfdecisionfunctionxyreshapeXXshape
plot decision boundary and margins
axcontourXX YY Z colorsk levels1 0 1 alpha05
linestyles
plot support vectors
axscatterclfsupportvectors 0 clfsupportvectors 1 s100
linewidth1 facecolorsnone edgecolorsk
pltshow
Total running time of the script 0 minutes 0021 seconds
Note Click here to download the full example code
5273 SVM (Support Vector Machine) with custom kernel
Simple usage of Support Vector Machines to classify a sample It will plot the decision surface and the support vectors
printdoc
import numpy as np
1402 Chapter 5 Examples
```


scikitlearn user guide Release 0213

```
import matplotlib.pyplot as plt
from sklearn import svm datasets
import some data to play with
iris datasetsloadiris
X irisdata 2 we only take the first two features We could
avoid this ugly slicing by using a twodim dataset
Y iristarget
defmykernelX Y
```

We create a custom kernel

```
2 0
kX Y X YT
0 1

M nparray2 0 0 10
returnnpdotnpdotX M YT
h 02 step size in the mesh
we create an instance of SVM (Support Vector Machine) and fit out data
clf svmSVCkernelmykernel
clffitX Y
Plot the decision boundary For that we will assign a color to each
point in the mesh xmin xmaxxmin ymax
xmin xmax X 0min 1 X 0max 1
ymax ymax X 1min 1 X 1max 1
xx yy npmeshgridnparangexmin xmax h nparangeymax h
Z clfpredictnpcxxravel yyravel
Put the result into a color plot
Z Zreshapexxshape
pltcolormeshxx yy Z cmappltcmPaired
Plot also the training points
pltscatterX 0 X 1 cY cmappltcmPaired edgecolorsk
plttitle3Class classification using Support Vector Machine with custom
kernel
pltaxistight
pltshow
```

Total running time of the script 0 minutes 0110 seconds

Note Click here to download the full example code

5274 SVM (Support Vector Machine) Weighted samples

Plot decision function of a weighted dataset where the size of points is proportional to its weight

The sample weighting rescales the C parameter which means that the classifier puts more emphasis on getting these

527 Support Vector Machines 1403

```
scikitlearn user guide Release 0213
points right The effect might often be subtle To emphasize the effect here we particularly weight outliers making
the deformation of the decision boundary very visible
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
defplotdecisionfunctionclassifier sampleweight axis title
    plot the decision function
xx yy npmeshgridnplinspace4 5 500 nplinspace4 5 500
Z classifierdecisionfunctionnpcxxravel yy.ravel
Z Z.reshapexxshape
    plot the line the points and the nearest vectors to the plane
axiscontourfxx yy Z alpha075 cmappltcmdbone
axisscatterX 0 X 1 cy s100 sampleweight alpha09
cmappltcmdbone edgecolorsblack
axisaxisoff
axissettitletitle
    we create 20 points
nprandomseed0
X nprnprandomrandn10 2 1 1 nprandomrandn10 2
y 1 10 1 10
sampleweightlastten absnprandomrandnlenX
sampleweightconstant np.oneslenX
    and bigger weights to some outliers
sampleweightlastten15 5
sampleweightlastten9 15
    for reference first fit without sample weights
    fit the model
1404 Chapter 5 Examples
```

scikitlearn user guide Release 0213

clfweights svmSVCgamma1

clfweightsfitX y sampleweightsampleweightlastten

clfnoweights svmSVCgamma1

clfnoweightsfitX y

fig axes pltsubplots1 2 figsize14 6

plotdecisionfunctionclfnoweights sampleweightconstant axes0

Constant weights

plotdecisionfunctionclfweights sampleweightlastten axes1

Modified weights

pltshow

Total running time of the script 0 minutes 0349 seconds

Note Click here to download the full example code

5275 SVM (Support Vector Machine) Separating hyperplane for unbalanced classes

Find the optimal separating hyperplane using an SVC (Support Vector Classification) for classes that are unbalanced

We first find the separating plane with a plain SVC (Support Vector Classification) and then plot dashed the separating hyperplane with automatic

correction for unbalanced classes

Note This example will also work by replacing SVCkernellinear with

SGDClassifierlosshinge Setting the loss parameter of the SGDClassifier equal tohinge will

yield behaviour such as that of a SVC (Support Vector Classification) with a linear kernel

For example try instead of the SVC (Support Vector Classification)

clf SGDClassifierniter100 alpha001

527 Support Vector Machines 1405

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn.datasets import makeblobs
    we create two clusters of random points
nsamples1 1000
nsamples2 100
centers 00 00 20 20
clustersstd 15 05
X y makeblobsnsamplesnsamples1 nsamples2
centerscenters
clusterstdclustersstd
randomstate0 shuffleFalse
    fit the model and get the separating hyperplane
clf svmSVCKernellinear C10
clffitX y
    fit the model and get the separating hyperplane using weighted classes
wclf svmSVCKernellinear classweight1 10
wclffitX y
    plot the samples
1406 Chapter 5 Examples
```

scikitlearn user guide Release 0213

pltscatterX 0 X 1 cy cmappltcmPaired edgecolorsk

plot the decision functions for both classifiers

ax pltgca

xlim axgetxlim

ylim axgetylim

create grid to evaluate model

xx nplinspacexlim0 xlim1 30

yy nplinspaceylim0 ylim1 30

YY XX npmeshgridyy xx

xy npvstackXXravel YYravelT

get the separating hyperplane

Z clfdecisionfunctionxyreshapeXXshape

plot decision boundary and margins

a axcontourXX YY Z colorsk levels0 alpha05 linestyle

get the separating hyperplane for weighted classes

Z wclfdecisionfunctionxyreshapeXXshape

plot decision boundary and margins for weighted classes

b axcontourXX YY Z colorsr levels0 alpha05 linestyle

pltlegendacollections0 bcollections0 non weighted weighted

locupper right

pltshow

Total running time of the script 0 minutes 0034 seconds

Note Click here to download the full example code

5276 SVMKernels

Three different types of SVMKernels are displayed below The polynomial and RBF (Radial Basis Function) are especially useful when the datapoints are not linearly separable

527 Support Vector Machines 1407

scikitlearn user guide Release 0213

-
-

1408 Chapter 5 Examples

scikitlearn user guide Release 0213

```
•
printdoc
Code source Gaël Varoquaux
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
Our dataset and targets
X npc4 7
15 1
14 9
13 12
11 2
12 4
5 12
15 21
1 1

13 8
12 5
2 2
5 24
2 23
0 27
13 21T
Y 0 8 1 8
figure number
fignum 1
fit the model
forkernelinlinear poly rbf
clf svmSVCKernelkernel gamma2
clffitX Y
527 Support Vector Machines 1409
```

scikitlearn user guide Release 0213

plot the line the points and the nearest vectors to the plane

pltfigurefignum figsize4 3

pltclf

pltscatterclfsupportvectors 0 clfsupportvectors 1 s80

facecolorsnone zorder10 edgecolorsk

pltscatterX 0 X 1 cY zorder10 cmappltcmPaired

edgecolorsk

pltaxistight

xmin 3

xmax 3

ymin 3

ymax 3

XX YY npmgridxminxmax200j yminymax200j

Z clfdecisionfunctionnpcXXravel YYravel

Put the result into a color plot

Z ZreshapeXXshape

pltfigurefignum figsize4 3

pltcolormeshXX YY Z 0 cmappltcmPaired

pltcontourXX YY Z colorsk k k linestyles

levels5 0 5

pltxlimxmin xmax

pltylimymin ymax

pltxticks

pltyticks

fignum fignum 1

pltshow

Total running time of the script 0 minutes 0096 seconds

Note Click here to download the full example code

5277 SVMAnova SVM (Support Vector Machine) with univariate feature selection

This example shows how to perform univariate feature selection before running a SVC (Support Vector Classification) support vector classifier to improve the classification scores We use the iris dataset 4 features and add 36 noninformative features We can find that our model achieves best performance when we select around 10 of features

1410 Chapter 5 Examples


```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import loadiris
from sklearn.feature_selection import SelectPercentile chi2
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC (Support Vector Classification)
```

```
    Import some data to play with
X, y = loadiris(return_Xy=True)
    Add noninformative features
np.random.seed(0)
X = np.hstack((X, 2 * np.random.random(X.shape[0: 36])))
```

```
    Create a feature selection transform, a scaler, and an instance of SVM (Support Vector Machine) that we
    combine together to have a full-blown estimator
clf = Pipeline([
    ('anova', SelectPercentile(chi2)),
    ('scaler', StandardScaler()),
    ('svc', SVC(gamma=auto))
])
527 Support Vector Machines 1411
```

```
Plot the crossvalidation score as a function of percentile of features
scoremeans list
scorestds list
percentiles 1 3 6 10 15 20 30 40 60 80 100
forpercentile inpercentiles
clfsetparamsanovapercentilepercentile
thisscores crossvalscoreclf X y cv5
scoremeansappendthisscoresmean
scorestdsappendthisscoresstd
plterrorbarpercentiles scoremeans nparrayscorestds
plttitle
Performance of the SVMAnova varying the percentile of features selected
pltxticksnplinspace0 100 11 endpointTrue
pltxlabelPercentile
pltylabelAccuracy Score
pltaxistight
pltshow
Total running time of the script 0 minutes 0199 seconds
Note Click here to download the full example code
5278 Support Vector Regression SVR (Support Vector Regression) using linear and nonlinear kernels
Toy example of 1D regression using linear polynomial and RBF (Radial Basis Function) kernels
1412 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
from sklearnsvm import SVR (Support Vector Regression)
import matplotlib.pyplot as plt

Generate sample data
X np.sort(5 * np.random.rand(40, 1) * axis0)
y np.sin(X * ravel)

Add noise to targets
y[5: 30] += np.random.rand(8)

Fit regression model
svrrbf SVR(kernel=rbf, C=100, gamma=0.1, epsilon=1)
svrlin SVR(kernel=linear, C=100, gamma=auto)
svrpoly SVR(kernel=poly, C=100, gamma=auto, degree=3, epsilon=1)
coef0=1

Look at the results
lw = 2
svrs = svrrbf, svrlin, svrpoly
kernel_label = RBF (Radial Basis Function) Linear Polynomial
model_color = m, c, g
527 Support Vector Machines 1413
```

```
scikitlearn user guide Release 0213
fig axes pltsubplotsnrows1 ncols3 figsize15 10 shareyTrue
forix svr inenumeratesvrs
axesixplotX svrfiT X ypredictX colormodelcolorix lwlw
label modelformatkernellabelix
axesixscatterXsvrsupport ysvrsupport facecolornone
edgecolormodelcolorix s50
label support vectorsformatkernellabelix
axesixscatterXnpsetdiff1dnparangelenX svrsupport
ynpsetdiff1dnparangelenX svrsupport
facecolornone edgecolork s50
labelother training data
axesixlegendlocupper center bboxtoanchor05 11
ncol1 fancyboxTrue shadowTrue
figtext05 004 data hacenter vacenter
figtext006 05 target hacenter vacenter rotationvertical
figsuptitleSupport Vector Regression fontsize14
pltshow
Total running time of the script 0 minutes 3104 seconds
Note Click here to download the full example code
5279 SVM (Support Vector Machine) Margins Example
The plots below illustrate the effect the parameter C has on the separation line A large value of C basically tells our
model that we do not have that much faith in our data's distribution and will only consider points close to line of
separation
A small value of C includes more all the observations allowing the margins to be calculated using all the data in the
area
•
1414 Chapter 5 Examples
```

scikitlearn user guide Release 0213

•

```
printdoc
Code source Gaël Varoquaux
Modified for documentation by Jaques Grobler
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm

# we create 40 separable points
np.random.seed(0)
X = np.random.randn(20, 2)
Y = [0] * 20
X = np.concatenate((X, np.random.randn(20, 2)), axis=0)
Y = [1] * 20

# figure number
fig = plt.figure(1)

# fit the model
clf = svm.SVC(kernel='linear', C=1)
clf.fit(X, Y)

# get the separating hyperplane
w = clf.coef_0
a = w[0]
xx = np.linspace(-5, 5)
yy = -a * x / w[1]

# plot the parallels to the separating hyperplane that pass through the
# support vectors margin away from hyperplane in direction
# perpendicular to hyperplane This is sqrt(1+a^2) away vertically in
# 2d
margin = 1 / np.sqrt(1 + w[0]**2)
yydown = yy - margin
yyup = yy + margin

527 Support Vector Machines 1415
```

```
scikitlearn user guide Release 0213
plot the line the points and the nearest vectors to the plane
pltfigurefignum figsize4 3
pltclf
pltplotxx yy k
pltplotxx yydown k
pltplotxx yyup k
pltscatterclfsupportvectors 0 clfsupportvectors 1 s80
facecolorsnone zorder10 edgecolorsk
pltscatterX 0 X 1 cY zorder10 cmappltcmPaired
edgecolorsk
pltaxistight
xmin 48
xmax 42
ymin 6
ymax 6
XX YY npmgridxminxmax200j yminymax200j
Z clfpredictnpcXXravel YYravel
Put the result into a color plot
Z ZreshapeXXshape
pltfigurefignum figsize4 3
pltcolormeshXX YY Z cmappltcmPaired
pltxlimxmin xmax
pltylimymin ymax
pltxticks
pltyticks
fignum fignum 1
pltshow
Total running time of the script 0 minutes 0061 seconds
Note Click here to download the full example code
52710 Oneclass SVM (Support Vector Machine) with nonlinear kernel RBF (Radial Basis Function)
An example using a oneclass SVM (Support Vector Machine) for novelty detection
Oneclass SVM (Support Vector Machine) is an unsupervised algorithm that learns a decision function for novelty detection classifying new
data as similar or different to the training set
1416 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
import matplotlibfontmanager
from sklearn import svm
xx yy npmeshgridnplinspace5 5 500 nplinspace5 5 500
Generate train data
X 03 nprandomrandn100 2
Xtrain nprX 2 X 2
Generate some regular novel observations
X 03 nprandomrandn20 2
Xtest nprX 2 X 2
Generate some abnormal novel observations
Xoutliers nprandomuniformlow4 high4 size20 2
fit the model
clf svmOneClassSVMnu01 kernelrbf gamma01
clffitXtrain
ypredtrain clfpredictXtrain
ypredtest clfpredictXtest
ypredoutliers clfpredictXoutliers
nerrortrain ypredtrainypredtrain 1size
nerrortest ypredtestypredtest 1size
nerroroutliers ypredoutliersypredoutliers 1size
527 Support Vector Machines 1417
```

scikitlearn user guide Release 0213

plot the line the points and the nearest vectors to the plane

Z clfdecisionfunctionnpcxxravel yy.ravel

Z Zreshapexxshape

plttitleNovelty Detection

pltcontourfxx yy Z levelsnplinspaceZmin 0 7 cmappltcmPuBu

a pltcontourxx yy Z levels0 linewidths2 colorsdarkred

pltcontourfxx yy Z levels0 Zmax colorspalevioletred

s 40

b1 pltscatterXtrain 0 Xtrain 1 cwhite ss edgecolorsk

b2 pltscatterXtest 0 Xtest 1 cblueviolet ss

edgecolorsk

c pltscatterXoutliers 0 Xoutliers 1 cgold ss

edgecolorsk

pltaxistight

pltxlim5 5

pltylim5 5

pltlegendacollections0 b1 b2 c

learned frontier training observations

new regular observations new abnormal observations

locupper left

propmatplotlibfontmanagerFontPropertiessize11

pltxlabel

error train d200 errors novel regular d40

errors novel abnormal d40

nerrortrain nerrortest nerroroutliers

pltshow

Total running time of the script 0 minutes 0196 seconds

Note Click here to download the full example code

52711 Plot different SVM (Support Vector Machine) classifiers in the iris dataset

Comparison of different linear SVM (Support Vector Machine) classifiers on a 2D projection of the iris dataset We only consider the first 2 features of this dataset

- Sepal length
- Sepal width

This example shows how to plot the decision surface for four SVM (Support Vector Machine) classifiers with different kernels

The linear models LinearSVC andSVCKernellinear yield slightly different decision boundaries

This can be a consequence of the following differences

- LinearSVC minimizes the squared hinge loss while SVC (Support Vector Classification) minimizes the regular hinge loss
- LinearSVC uses the OnevsAll also known as OnevsRest multiclass reduction while SVC (Support Vector Classification) uses the One vsOne multiclass reduction

Both linear models have linear decision boundaries intersecting hyperplanes while the nonlinear kernel models polynomial or Gaussian RBF (Radial Basis Function) have more flexible nonlinear decision boundaries with shapes that depend on the kind of kernel and its parameters

1418 Chapter 5 Examples

scikitlearn user guide Release 0213

Note while plotting the decision function of classifiers for toy 2D datasets can help get an intuitive understanding of their respective expressive power be aware that those intuitions don't always generalize to more realistic high dimensional problems

```
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm datasets
defmakemeshgridx y h02
Create a mesh of points to plot in
Parameters
```

x data to base xaxis meshgrid on
y data to base yaxis meshgrid on
h stepsize for meshgrid optional
Returns

xx yy ndarray

scikitlearn user guide Release 0213

xmin xmax xmin 1 xmax 1

ymin ymax ymin 1 ymax 1

xx yy npmeshgridnparangexmin xmax h

nparangeymin ymax h

returnxx yy

defplotcontoursax clf xx yy params

Plot the decision boundaries for a classifier

Parameters

ax matplotlib axes object

clf a classifier

xx meshgrid ndarray

yy meshgrid ndarray

params dictionary of params to pass to contourf optional

Z clfpredictnpcxxravel yyravel

Z Zreshapexxshape

out axcontourfxx yy Z params

returnout

import some data to play with

iris datasetsloadiris

Take the first two features We could avoid this by using a twodim dataset

X irisdata 2

y iristarget

we create an instance of SVM (Support Vector Machine) and fit out data We do not scale our

data since we want to plot the support vectors

C 10 SVM (Support Vector Machine) regularization parameter

models svmSVCKernellinear CC

svmLinearSVCCC maxiter10000

svmSVCKernelrbf gamma07 CC

svmSVCKernelpoly degree3 gammaauto CC

models clffitX y forclfinmodels

title for the plots

titles SVC (Support Vector Classification) with linear kernel

LinearSVC linear kernel

SVC (Support Vector Classification) with RBF (Radial Basis Function) kernel

SVC (Support Vector Classification) with polynomial degree 3 kernel

Setup 2x2 grid for plotting

fig sub pltsubplots2 2

pltsubplotsadjustwspace04 hspace04

X0 X1 X 0 X 1

xx yy makemeshgridX0 X1

forclf title ax inzipmodels titles subflatten

plotcontoursax clf xx yy

cmappltcmcoolwarm alpha08

axscatterX0 X1 cy cmappltcmcoolwarm s20 edgecolorsk

axsetxlimxxmin xxmax

axsetylimyymin yymax

1420 Chapter 5 Examples

scikitlearn user guide Release 0213

axsetxlabelSepal length

axsetylabelSepal width

axsetxticks

axsetyticks

axsettitletitle

pltshow

Total running time of the script 0 minutes 0481 seconds

Note Click here to download the full example code

52712 Scaling the regularization parameter for SVCs

The following example illustrates the effect of scaling the regularization parameter when using Support Vector Machines for classification. For SVC (Support Vector Classification) classification we are interested in a risk minimization for the equation

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \max(0, 1 - y_i \mathbf{w} \cdot \mathbf{x}_i)$$

where

- $\|\mathbf{w}\|^2$ is used to set the amount of regularization
- \mathcal{L} is a loss function of our samples and our model parameters
-

is a penalty function of our model parameters

If we consider the loss function to be the individual error per sample then the data fit term or the sum of the error for each sample will increase as we add more samples. The penalization term however will not increase.

When using for example cross validation to set the amount of regularization with C there will be a different amount of samples between the main problem and the smaller problems within the folds of the cross validation.

Since our loss function is dependent on the amount of samples the latter will influence the selected value of C . The question that arises is: How do we optimally adjust C to account for the different amount of training samples?

The figures below are used to illustrate the effect of scaling our C to compensate for the change in the number of samples in the case of using an l_1 penalty as well as the l_2 penalty.

l_1 penalty case

In the l_1 case theory says that prediction consistency (i.e. that under given hypothesis the estimator learned predicts as well as a model knowing the true distribution) is not possible because of the bias of the l_1 . It does say however that model consistency in terms of finding the right set of nonzero parameters as well as their signs can be achieved by scaling C .

l_2 penalty case

The theory says that in order to achieve prediction consistency the penalty parameter should be kept constant as the number of samples grow.

527 Support Vector Machines 1421

Simulations

The two figures below plot the values of C on the x-axis and the corresponding crossvalidation scores on the y-axis for several different fractions of a generated dataset

In the l_1 penalty case the crossvalidation error correlates best with the test error when scaling our C with the number of samples n which can be seen in the first figure

For the l_2 penalty case the best result comes from the case where C is not scaled

Note

Two separate datasets are used for the two different plots The reason behind this is the l_1 case works better on sparse data while l_2 is better suited to the non sparse case

1422 Chapter 5 Examples

scikitlearn user guide Release 0213

-

527 Support Vector Machines 1423

```
scikitlearn user guide Release 0213
•
printdoc
Author Andreas Mueller amuelleraisunibonnde
Jaques Grobler jaquesgroblerinriafr
License BSD (Binary Search Decision) 3 clause
import numpy as np
import matplotlib.pyplot as plt
from sklearnsvm import LinearSVC
from sklearnmodelselection import ShuffleSplit
1424 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
from sklearn.model_selection import GridSearchCV
from sklearn.utils import check_random_state
from sklearn import datasets
rnd = check_random_state(1)
# set up dataset
nsamples = 100
nfeatures = 300
# I1 data only 5 informative features
X1, y1 = datasets.make_classification(nsamples, nfeatures,
                                     nfeatures - ninformative, 5,
                                     random_state=1)
# I2 data non sparse but less features
y2 = np.sign(5 * rnd.randn(samples))
X2 = rnd.randn(nsamples, nfeatures - 5)
X2 = np.hstack((X2, y2.reshape((-1,))))
clfsets = LinearSVC(penalty=l1, loss=squared_hinge, dual=False,
                    tol=1e-3)
nplogspace(2, 3, 10, X1, y1)
LinearSVC(penalty=l2, loss=squared_hinge, dual=True,
          tol=1e-4)
nplogspace(4, 5, 2, 10, X2, y2)
colors = ['navy', 'cyan', 'darkorange']
lw = 2
for clf, cs, X, y, in clfsets:
    # set up the plot for each regressor
    fig, axes = plt.subplots(nrows=2, sharey=True, figsize=(9, 10))
    for k, trainsize in enumerate(np.linspace(0.3, 0.7, 31)):
        paramgrid = dict(Cs=cs)
        # To get nice curve we need a large number of iterations to
        # reduce the variance
        grid = GridSearchCV(clf, paramgrid, refit=False,
                            cv=ShuffleSplit(trainsize, trainsize,
                                             testsize=3,
                                             n_splits=250,
                                             random_state=1))
        scores = grid.cv_results_.mean_test_score
        scales = 1 / NoScaling
        nsamples, trainsize = 1, nsamples

    for ax, scaler_name in zip(axes, scales):
        ax.set_xlabel(C)
        ax.set_ylabel(CV Score)
        gridcs = cs * float(scaler)
        # scale the Cs
        ax.semilogx(gridcs, scores, label=fraction * 2f)
        trainsize, color, colorsk, lw, lw
        ax.set_title(f'scaling: s penalty: s loss: s')
        name = clf.penalty, clf.loss
527 Support Vector Machines 1425
```

scikitlearn user guide Release 0213

pltlegendlocbest

pltshow

Total running time of the script 0 minutes 14575 seconds

Note Click here to download the full example code

52713 RBF (Radial Basis Function) SVM (Support Vector Machine) parameters

This example illustrates the effect of the parameters gamma and C of the Radial Basis Function RBF (Radial Basis Function) kernel SVM (Support Vector Machine). Intuitively the gamma parameter defines how far the influence of a single training example reaches with low values meaning ‘far’ and high values meaning ‘close’. The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.

The C parameter trades off correct classification of training examples against maximization of the decision function’s margin. For larger values of C a smaller margin will be accepted if the decision function is better at classifying all training points correctly. A lower C will encourage a larger margin therefore a simpler decision function at the cost of training accuracy. In other words ‘C’ behaves as a regularization parameter in the SVM (Support Vector Machine). The first plot is a visualization of the decision function for a variety of parameter values on a simplified classification problem involving only 2 input features and 2 possible target classes binary classification. Note that this kind of plot is not possible to do for problems with more features or target classes.

The second plot is a heatmap of the classifier’s crossvalidation accuracy as a function of C and gamma. For this example we explore a relatively large grid for illustration purposes. In practice a logarithmic grid from 10⁻³ to 10³ is usually sufficient. If the best parameters lie on the boundaries of the grid it can be extended in that direction in a subsequent search.

Note that the heat map plot has a special colorbar with a midpoint value close to the score values of the best performing models so as to make it easy to tell them apart in the blink of an eye.

The behavior of the model is very sensitive to the gamma parameter. If gamma is too large the radius of the area of influence of the support vectors only includes the support vector itself and no amount of regularization with C will be able to prevent overfitting.

When gamma is very small the model is too constrained and cannot capture the complexity or “shape” of the data. The region of influence of any selected support vector would include the whole training set. The resulting model will behave similarly to a linear model with a set of hyperplanes that separate the centers of high density of any pair of two classes.

For intermediate values we can see on the second plot that good models can be found on a diagonal of C and gamma. Smooth models lower gamma values can be made more complex by increasing the importance of classifying each point correctly larger C values hence the diagonal of good performing models.

Finally one can also observe that for some intermediate values of gamma we get equally performing models when C becomes very large. It is not necessary to regularize by enforcing a larger margin. The radius of the RBF (Radial Basis Function) kernel alone acts as a good structural regularizer. In practice though it might still be interesting to simplify the decision function with a lower value of C so as to favor models that use less memory and that are faster to predict.

We should also note that small differences in scores results from the random splits of the crossvalidation procedure. Those spurious variations can be smoothed out by increasing the number of CV (Cross-Validation) iterations n_splits at the expense of compute time. Increasing the value number of C_range and gamma_range steps will increase the resolution of the hyperparameter heat map.

1426 Chapter 5 Examples

scikitlearn user guide Release 0213

-

527 Support Vector Machines 1427

scikitlearn user guide Release 0213

•

Out

The best parameters are C 10 gamma 01 with a score of 097

printdoc

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.colors import Normalize

from sklearn.svm import SVC (Support Vector Classification)

from sklearn.preprocessing import StandardScaler

from sklearn.datasets import loadiris

from sklearn.model_selection import StratifiedShuffleSplit

from sklearn.model_selection import GridSearchCV

Utility function to move the midpoint of a colormap to be around
the values of interest

class MidpointNormalize Normalize

1428 Chapter 5 Examples

```
scikitlearn user guide Release 0213
definitself vminNone vmaxNone midpointNone clipFalse
selfmidpoint midpoint
Normalizewithself vmin vmax clip
defcallself value clipNone
x y selfvmin selfmidpoint selfvmax 0 05 1
returnnp.ma.masked_array(np.interp(value, x, y
```

Load and prepare data set

```
dataset for grid search
iris loadiris
X irisdata
y iristarget
Dataset for decision function visualization we only keep the first two
features in X and subsample the dataset to keep only 2 classes and
make it a binary classification problem
X2d X 2
X2d X2dy 0
y2d yy 0
y2d 1
```

```
It is usually a good idea to scale the data for SVM (Support Vector Machine) training
We are cheating a bit in this example in scaling all of the data
instead of fitting the transformation on the training set and
just applying it on the test set
scaler StandardScaler
X scaler.fit_transform(X)
X2d scaler.fit_transform(X2d)
```

Train classifiers

```
For an initial search a logarithmic grid with basis
10 is often helpful Using a basis of 2 a finer
tuning can be achieved but at a much higher cost
Crange np.logspace(2, 10, 13)
gamma_range np.logspace(-9, 3, 13)
param_grid dict(gamma=gamma_range, C=Crange)
cv StratifiedShuffleSplit(n_splits=5, test_size=0.2, random_state=42)
grid GridSearchCV(SVC(), param_grid, cv=cv)
grid.fit(X, y)
print("The best parameters are with a score of %2f" %
      grid.best_params_)
grid.best_params_
grid.best_score_
Now we need to fit a classifier for all parameters in the 2d version
we use a smaller set of parameters here because it takes a while to train
C2d_range 1e-2 1 1e2
gamma2d_range 1e-1 1 1e1
classifiers
527 Support Vector Machines 1429
```

scikitlearn user guide Release 0213

```
forCinC2drange
forgammaingamma2drange
clf SVCCC gammagama
clffitX2d y2d
classifiersappendC gamma clf
```

Visualization

```
draw visualization of parameter effects
pltfigurefigsize8 6
xx yy npmeshgridnplinspace3 3 200 nplinspace3 3 200
fork C gamma clf inenumerateclassifiers
    evaluate decision function in a grid
Z clfdecisionfunctionnpcxxravel yyravel
Z Zreshapexxshape
    visualize decision function for these parameters
pltsubplotlenC2drange lengamma2drange k 1
plttitlegamma10 d C10d nplog10gamma nplog10C
sizemedium
    visualize parameters effect on decision function
pltpcolormeshxx yy Z cmappltcmRdBu
pltscatterX2d 0 X2d 1 cy2d cmappltcmRdBur
edgecolorsk
pltxticks
pltyticks
pltaxistight
scores_gridcvresultsmeantestscorereshapelenCrange
lengammarange
    Draw heatmap of the validation accuracy as a function of gamma and C
```

The score are encoded as colors with the hot colormap which varies from dark red to bright yellow As the most interesting scores are all located in the 092 to 097 range we use a custom normalizer to set the midpoint to 092 so as to make it easier to visualize the small variations of score values in the interesting range while not brutally collapsing all the low score values to the same color

```
pltfigurefigsize8 6
pltsubplotsadjustleft2 right095 bottom015 top095
pltimshowscores interpolationnearest cmappltcmhot
normMidpointNormalizevmin02 midpoint092
pltxlabelgamma
pltylabelC
pltcolorbar
pltxticksnparangelengammarange gammarange rotation45
plttyticksnparangelenCrange Crange
plttitleValidation accuracy
pltshow
```

Total running time of the script 0 minutes 3597 seconds
1430 Chapter 5 Examples

scikitlearn user guide Release 0213

528 Working with text documents

Examples concerning the sklearnfeatureextractiontext module

Note Click here to download the full example code

5281 FeatureHasher and DictVectorizer Comparison

Compares FeatureHasher and DictVectorizer by using both to vectorize text documents

The example demonstrates syntax and speed only it doesn't actually do anything useful with the extracted vectors

See the example scripts documentclassification20newsgroupsclusteringpy for actual learning on text documents

A discrepancy between the number of terms reported for DictVectorizer and for FeatureHasher is to be expected due to hash collisions

Out

Usage homecirclecipprojectexampletextplothashingvsdictvectorizerpy n

↪featuresforhashing

The default number of features is 2 18

Loading 20 newsgroups training data

3803 documents 6245MB

DictVectorizer

done in 0979095s at 6378MBs

Found 47928 unique terms

FeatureHasher on frequency dicts

done in 0816599s at 7647MBs

Found 43873 unique terms

FeatureHasher on raw tokens

done in 0935579s at 6675MBs

Found 43873 unique terms

Author Lars Buitinck

License BSD (Binary Search Decision) 3 clause

```
from collections import defaultdict
import re
import sys
from time import time
import numpy as np
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction import DictVectorizer FeatureHasher
```

528 Working with text documents 1431

scikitlearn user guide Release 0213

```
defnnonzerocolumnsX
```

Returns the number of nonzero columns in a CSR (Compressed Sparse Row) matrix X

```
returnlennpuniqueXnonzero1
```

```
deftokensdoc
```

Extract tokens from doc

This uses a simple regex to break strings into tokens For a more principled approach see CountVectorizer or TfidfVectorizer

```
returntoklower fortokinrefindallrw doc
```

```
deftokenfreqsdoc
```

Extract a dict mapping tokens from doc to their frequencies

```
freq defaultdictint
```

```
fortokintokensdoc
```

```
freqtok 1
```

```
returnfreq
```

categories

altatheism

compgraphics

compsysibmpchardware

miscforsale

recautos

scispace

talkreligionmisc

Uncomment the following line to use a larger set 11k documents

```
categories None
```

```
printdoc
```

```
printUsage snfeaturesforhashing sysargv0
```

```
print The default number of features is 2 18
```

```
print
```

```
try
```

```
nfeatures intsysargv1
```

```
exceptIndexError
```

```
nfeatures 2 18
```

```
exceptValueError
```

```
printnot a valid number of features r sysargv1
```

```
sysexit1
```

```
printLoading 20 newsgroups training data
```

```
rawdata fetch20newsgroupssubsettrain categoriescategoriesdata
```

```
datasizemb sumlensencodeutf8 forsinrawdata 1e6
```

```
printddocuments 03fMB lenrawdata datasizemb
```

```
print
```

```
printDictVectorizer
```

```
t0 time
```

```
vectorizer DictVectorizer
```

```
vectorizerfittransformtokenfreqsd fordinrawdata
```

1432 Chapter 5 Examples

scikitlearn user guide Release 0213

```
duration time t0
printdone in fs at03fMBs duration datasizemb duration
printFound dunique terms lenvectorizergetfeaturenames
print
printFeatureHasher on frequency dicts
t0 time
hasher FeatureHashernfeaturesnfeatures
X hashertransformtokenfreqsd fordinrawdata
duration time t0
printdone in fs at03fMBs duration datasizemb duration
printFound dunique terms nnonzerocolumnsX
print
printFeatureHasher on raw tokens
t0 time
hasher FeatureHashernfeaturesnfeatures inputtypestring
X hashertransformtokensd fordinrawdata
duration time t0
printdone in fs at03fMBs duration datasizemb duration
printFound dunique terms nnonzerocolumnsX
Total running time of the script 0 minutes 3036 seconds
```

Note Click here to download the full example code

5282 Clustering text documents using kmeans

This is an example showing how the scikitlearn can be used to cluster documents by topics using a bagofwords approach This example uses a scipysparse matrix to store the features instead of standard numpy arrays

Two feature extraction methods can be used in this example

- TfidfVectorizer uses a inmemory vocabulary a python dict to map the most frequent words to features indices and hence compute a word occurrence frequency sparse matrix The word frequencies are then reweighted using the Inverse Document Frequency IDF vector collected featurewise over the corpus
- HashingVectorizer hashes word occurrences to a fixed dimensional space possibly with collisions The word count vectors are then normalized to each have l2norm equal to one projected to the euclidean unitball which seems to be important for kmeans to work in high dimensional space

HashingVectorizer does not provide IDF weighting as this is a stateless model the fit method does nothing

When IDF weighting is needed it can be added by pipelining its output to a TfidfTransformer instance

Two algorithms are demoed ordinary kmeans and its more scalable cousin minibatch kmeans

Additionally latent semantic analysis can also be used to reduce dimensionality and discover latent patterns in the data

It can be noted that kmeans and minibatch kmeans are very sensitive to feature scaling and that in this case the IDF weighting helps improve the quality of the clustering by quite a lot as measured against the “ground truth” provided by the class label assignments of the 20 newsgroups dataset

This improvement is not visible in the Silhouette Coefficient which is small for both as this measure seem to suffer from the phenomenon called “Concentration of Measure” or “Curse of Dimensionality” for high dimensional datasets

scikitlearn user guide Release 0213

such as text data Other measures such as Vmeasure and Adjusted Rand Index are information theoretic based evaluation scores as they are only based on cluster assignments rather than distances hence not affected by the curse of dimensionality

Note as kmeans is optimizing a nonconvex objective function it will likely end up in a local optimum Several runs with independent random init might be necessary to get a good convergence

Out

Usage plotdocumentclusteringpy options

Options

h help show this help message and exit

IsaNCOMPONENTS Preprocess documents with latent semantic analysis

nominibatch Use ordinary kmeans algorithm in batch mode

noidf Disable Inverse Document Frequency feature weighting

usehashing Use a hashing feature vectorizer

nfeaturesNFEATURES

Maximum number of features dimensions to extract from text

verbose Print progress reports inside kmeans algorithm

Loading 20 newsgroups dataset for categories altatheism talkreligionmisc compgraphics scispace 3387 documents 4 categories

Extracting features from the training dataset using a sparse vectorizer done in 0691398s

nsamples 3387 nfeatures 10000

Clustering sparse data with MiniBatchKMeansbatchsize1000 initsize1000 n

↪clusters4 ninit1

verboseFalse

done in 0057s

Homogeneity 0359

Completeness 0440

Vmeasure 0396

Adjusted RandIndex 0253

Silhouette Coefficient 0007

Top terms per cluster

Cluster 0 henry alaska toronto moon zoo spencer aurora space nsmca zoology

Cluster 1 com graphics university posting host nntp know uk article cs

Cluster 2 god com sandvik people keith morality sgi kent livesey jesus

Cluster 3 space nasa access gov digex pat shuttle hst orbit net

Author Peter Prettenhofer peterprettenhofergmailcom

Lars Buitinck

License BSD (Binary Search Decision) 3 clause

```
from sklearndatasets import fetch20newsgroups
from sklearndecomposition import TruncatedSVD
from sklearnfeatureextractiontext import TfidfVectorizer
from sklearnfeatureextractiontext import HashingVectorizer
```

1434 Chapter 5 Examples


```
scikitlearn user guide Release 0213
from sklearnfeatureextractiontext import TfIdfTransformer
from sklearnpipeline import makepipeline
from sklearnpreprocessing import Normalizer
from sklearn import metrics
from sklearncluster import KMeans MiniBatchKMeans
import logging
from optparse import OptionParser
import sys
from time import time
import numpy as np
    Display progress logs on stdout
loggingbasicConfiglevelloggingINFO
format asctimes levelnames messages
    parse commandline arguments
op OptionParser
opaddoptionlsa
destncomponents typeint
helpPreprocess documents with latent semantic analysis
opaddoptionnominibatch
actionstorefalse destminibatch defaultTrue
helpUse ordinary kmeans algorithm in batch mode
opaddoptionnoidf
actionstorefalse destuseidf defaultTrue
helpDisable Inverse Document Frequency feature weighting
opaddoptionusehashing
actionstoretrue defaultFalse
helpUse a hashing feature vectorizer
opaddoptionnfeatures typeint default10000
helpMaximum number of features dimensions
    to extract from text
opaddoptionverbose
actionstoretrue destverbose defaultFalse
helpPrint progress reports inside kmeans algorithm
printdoc
opprinthelp
defisinteractive
return not hasattrsysmodulesmain file
    workaround for Jupyter notebook and IPython console
argv ifisinteractive elsesysargv1
opts args oparseargsargv
iflenargs 0
    operrorthis script takes no arguments
sysexit1

    Load some categories from the training set
528 Working with text documents 1435
```

scikitlearn user guide Release 0213

categories
altatheism
talkreligionmisc
compgraphics
scispace

Uncomment the following to do the analysis on all the categories

```
categories None
printLoading 20 newsgroups dataset for categories
printcategories
dataset fetch20newsgroupssubsetall categoriescategories
shuffleTrue randomstate42
printddocuments lendatasetdata
printdcategories lendatasettargetnames
print
labels datasettarget
truek npuniqueylabelsshape0
printExtracting features from the training dataset
using a sparse vectorizer
t0 time
ifoptsusehashing
ifoptsuseidf
    Perform an IDF normalization on the output of HashingVectorizer
hasher HashingVectorizernfeaturesoptsnfeatures
stopwordseenglish alternatesignFalse
normNone binaryFalse
vectorizer makepipelinehasher TfidfTransformer
else
vectorizer HashingVectorizernfeaturesoptsnfeatures
stopwordseenglish
alternatesignFalse norml2
binaryFalse
else
vectorizer TfidfVectorizernmaxdf05 maxfeaturesoptsnfeatures
mindf2 stopwordseenglish
useidfoptsuseidf
X vectorizerfittransformdatasetdata
printdone in fs time t0
printnsamples d nfeatures d Xshape
print
ifoptsncomponents
printPerforming dimensionality reduction using LSA
t0 time
    Vectorizer results are normalized which makes KMeans behave as
    spherical kmeans for better results Since LSASVD results are
    not normalized we have to redo the normalization
svd TruncatedSVDoptsncomponents
normalizer NormalizercopyFalse
lsa makepipelinesvd normalizer
X lsafittransformX
1436 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
printdone in fs time t0
explainedvariance svdexplainedvarianceratiosum
printExplained variance of the SVD step format
intexplainedvariance 100
print

Do the actual clustering
ifoptsminibatch
km MiniBatchKMeansnclusterstruek initkmeans ninit1
initsize1000 batchsize1000 verboseoptsverbose
else
km KMeansnclusterstruek initkmeans maxiter100 ninit1
verboseoptsverbose
printClustering sparse data with s km
t0 time
kmfitX
printdone in 03fs time t0
print
printHomogeneity 03f metricshomogeneityscorelabels kmlabels
printCompleteness 03f metricscompletenessscorelabels kmlabels
printVmeasure 03f metricsvmeasurescorelabels kmlabels
printAdjusted RandIndex 3f
metricsadjustedrandscorelabels kmlabels
printSilhouette Coefficient 03f
metricssilhouettescoreX kmlabels samplesize1000
print
if notoptsusehashing
printTop terms per cluster
ifoptsncomponents
originalspacecentroids svdinversetransformkmclustercenters
ordercentroids originalspacecentroidsargsort 1
else
ordercentroids kmclustercentersargsort 1
terms vectorizergetfeaturenames
foriinrangetruek
printCluster d i end
forindinordercentroids i 10
prints termsind end
print
Total running time of the script 0 minutes 1137 seconds
Note Click here to download the full example code
528 Working with text documents 1437
```

scikitlearn user guide Release 0213

5283 Classification of text documents using sparse features

This is an example showing how scikitlearn can be used to classify documents by topics using a bagofwords approach This example uses a scipysparse matrix to store the features and demonstrates various classifiers that can efficiently handle sparse matrices

The dataset used in this example is the 20 newsgroups dataset It will be automatically downloaded then cached

The bar plot indicates the accuracy training time normalized and test time normalized of each classifier

Out

Usage plotdocumentclassification20newsgroupspy options

Options

h help show this help message and exit

report Print a detailed classification report

chi2selectSELECTCHI2

Select some number of features using a chisquared

test

confusionmatrix Print the confusion matrix

top10 Print ten most discriminative terms per class for

every classifier

allcategories Whether to use all categories or not

usehashing Use a hashing vectorizer

nfeaturesNFEATURES

nfeatures when using the hashing vectorizer

filtered Remove newsgroup information that is easily overfit

headers signatures and quoting

1438 Chapter 5 Examples

scikitlearn user guide Release 0213
Loading 20 newsgroups dataset for categories
altatheism talkreligionmisc compgraphics scispace
data loaded
2034 documents 3980MB training set
1353 documents 2867MB test set
4 categories
Extracting features from the training data using a sparse vectorizer
done in 0412178s at 9655MBs
nsamples 2034 nfeatures 33809
Extracting features from the test data using the same vectorizer
done in 0351330s at 8162MBs
nsamples 1353 nfeatures 33809

Ridge Classifier

Training
RidgeClassifiersolversag tol001
train time 0132s
test time 0001s
accuracy 0896
dimensionality 33809
density 1000000

Perceptron

Training
Perceptronmaxiter50
train time 0017s
test time 0002s
accuracy 0888
dimensionality 33809
density 0255302

PassiveAggressive

Training
PassiveAggressiveClassifiermaxiter50
train time 0031s
test time 0002s
accuracy 0904
dimensionality 33809
density 0694674

kNN

Training
KNeighborsClassifiernneighbors10
train time 0002s
test time 0317s
528 Working with text documents 1439

scikitlearn user guide Release 0213
accuracy 0858

Random forest

Training
RandomForestClassifiernestimators100
train time 1671s
test time 0071s
accuracy 0840

L2 penalty

Training
LinearSVCdualFalse tol0001
train time 0145s
test time 0002s
accuracy 0900
dimensionality 33809
density 1000000

Training
SGDClassifiermaxiter50
train time 0030s
test time 0002s
accuracy 0902
dimensionality 33809
density 0579380

L1 penalty

Training
LinearSVCdualFalse penaltyl1 tol0001
train time 0301s
test time 0002s
accuracy 0873
dimensionality 33809
density 0005553

Training
SGDClassifiermaxiter50 penaltyl1
train time 0093s
test time 0002s
accuracy 0887
dimensionality 33809
density 0022901

ElasticNet penalty

scikitlearn user guide Release 0213
Training
SGDClassifiermaxiter50 penaltyelasticnet
train time 0252s
test time 0002s
accuracy 0899
dimensionality 33809
density 0187472

NearestCentroid aka Rocchio classifier

Training
NearestCentroid
train time 0004s
test time 0002s
accuracy 0855

Naive Bayes

Training
MultinomialNBalpha001
train time 0003s
test time 0001s
accuracy 0899
dimensionality 33809
density 1000000

Training
BernoulliNBalpha001
train time 0004s
test time 0003s
accuracy 0884
dimensionality 33809
density 1000000

Training
ComplementNBalpha01
train time 0004s
test time 0001s
accuracy 0911
dimensionality 33809
density 1000000

LinearSVC with L1based feature selection

Training
PipelineStepsFeaturesSelection
SelectFromModelEstimatorLinearSVCdualFalse penaltyl1
tol0001
classification LinearSVC
528 Working with text documents 1441

```
scikitlearn user guide Release 0213
train time 0252s
test time 0002s
accuracy 0880
  Author Peter Prettenhofer peterprettenhofergmailcom
  Olivier Grisel oliviergriselenstaorg
  Mathieu Blondel mathieumb blondelorg
  Lars Buitinck
  License BSD (Binary Search Decision) 3 clause
import logging
import numpy as np
from optparse import OptionParser
import sys
from time import time
import matplotlib.pyplot as plt
from sklearn.datasets import fetch20newsgroups
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import HashingVectorizer
from sklearn.feature_selection import SelectFromModel
from sklearn.feature_selection import SelectKBest chi2
from sklearn.linear_model import RidgeClassifier
from sklearn.pipeline import Pipeline
from sklearn.svm import LinearSVC
from sklearn.linear_model import SGDClassifier
from sklearn.linear_model import Perceptron
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.naive_bayes import BernoulliNB ComplementNB MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neighbors import NearestCentroid
from sklearn.ensemble import RandomForestClassifier
from sklearn.util import seq
from sklearn import metrics
  Display progress logs on stdout
logging.basicConfig(level=logging.INFO)
format = '%(asctime)s %(levelname)s %(message)s'
parser = argparse.ArgumentParser
op = parser.parse_args()
op.add_argument('--report',
action='store_true', dest='print_report',
help='Print a detailed classification report')
op.add_argument('--chi2select',
action='store_true', dest='select_chi2',
help='Select some number of features using a chi-squared test')
op.add_argument('--confusionmatrix',
action='store_true', dest='print_cm',
help='Print the confusion matrix')
op.add_argument('--top10',
action='store_true', dest='print_top10',
1442 Chapter 5 Examples
```



```
scikitlearn user guide Release 0213
helpPrint ten most discriminative terms per class
  for every classifier
opaddoptionallcategories
actionstoretrue destallcategories
helpWhether to use all categories or not
opaddoptionusehashing
actionstoretrue
helpUse a hashing vectorizer
opaddoptionnfeatures
actionstore typeint default2 16
helpnfeatures when using the hashing vectorizer
opaddoptionfiltered
actionstoretrue
helpRemove newsgroup information that is easily overfit
headers signatures and quoting
defisinteractive
return not hasattrsysmodulesmain file
  workaround for Jupyter notebook and IPython console
argv  ifisinteractive elsesysargv1
opts args  opparseargsargv
iflenargs  0
operrorthis script takes no arguments
sysexit1
printdoc
opprinthelp
print

  Load some categories from the training set
ifoptsallcategories
categories  None
else
categories
altatheism
talkreligionmisc
compgraphics
scispace

ifoptsfiltered
remove  headers footers quotes
else
remove
printLoading 20 newsgroups dataset for categories
printcategories ifcategories elseall
datatrain  fetch20newsgroupssubsettrain categoriescategories
shuffleTrue randomstate42
removeremove
datatest  fetch20newsgroupssubsettest categoriescategories
528 Working with text documents 1443
```

```
scikitlearn user guide Release 0213
shuffleTrue randomstate42
removerremove
printdata loaded
    order of labels in targetnames can be different from categories
targetnames datatraintargetnames
defsizebdocs
returnsumlensencodeutf8 forsindocs 1e6
datatrainsizeb sizebdatatraindata
datatestsizeb sizebdatatestdata
printdocuments 03fMB training set
lendatraindata datatrainsizeb
printdocuments 03fMB test set
lendatatestdata datatestsizeb
printcategories lentargetnames
print
    split a training set and a test set
ytrain ytest datatraintarget datatesttarget
printExtracting features from the training data using a sparse vectorizer
t0 time
ifoptsusehashing
vectorizer HashingVectorizerstopwordseenglish alternatesignFalse
nfeaturesoptsnfeatures
Xtrain vectorizertransformdatatraindata
else
vectorizer TfidfVectorizersublineartfTrue maxdf05
stopwordseenglish
Xtrain vectorizerfittransformdatatraindata
duration time t0
printdone in fs at03fMBs duration datatrainsizeb duration
printnsamples d nfeatures d Xtrainshape
print
printExtracting features from the test data using the same vectorizer
t0 time
Xtest vectorizertransformdatatestdata
duration time t0
printdone in fs at03fMBs duration datatestsizeb duration
printnsamples d nfeatures d Xtestshape
print
    mapping from integer feature name to original token string
ifoptsusehashing
featurenames None
else
featurenames vectorizergetfeaturenames
ifoptselectchi2
printExtracting dbest features by a chisquared test
optselectchi2
t0 time
1444 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
ch2 SelectKBestchi2 koptselectchi2
Xtrain ch2fittransformXtrain ytrain
Xtest ch2transformXtest
iffeaturenames
    keep selected feature names
featurenames featurenamesi fori
inch2getsupportindicesTrue
printdone in fs time t0
print
iffeaturenames
featurenames npasarrayfeaturenames
deftrims
Trim string to fit on terminal assuming 80column display
returnsiflens 80 elses77
```

```
Benchmark classifiers
defbenchmarkclf
print80
printTraining
printclf
t0 time
clffitXtrain ytrain
traintime time t0
printtrain time 03fs traintime
t0 time
pred clfpredictXtest
testtime time t0
printtest time 03fs testtime
score metricsaccuracycoreytest pred
printaccuracy 03f score
ifhasattrclf coef
printdimensionality d clfcoefshape1
printdensity f densityclfcoef
ifoptsprinttop10 andfeaturenames is notNone
printtop 10 keywords per class
fori label inenumeratetargetnames
top10 npargsortclfcoefi10
printtrimss label joinfeaturenamestop10
print
ifoptsprintreport
printclassification report
printmetricsclassificationreportytest pred
targetnamestargetnames
ifoptsprintcm
printconfusion matrix
printmetricsconfusionmatrixytest pred
528 Working with text documents 1445
```

```
scikitlearn user guide Release 0213
print
clfdescr  strclfsplit0
returnclfdescr score traintime testtime
results
forclf name in
RidgeClassifiertol1e2 solversag Ridge Classifier
Perceptronmaxiter50 tol1e3 Perceptron
PassiveAggressiveClassifiermaxiter50 tol1e3
PassiveAggressive
KNeighborsClassifiernneighbors10 kNN
RandomForestClassifiernestimators100 Random forest
print80
printname
resultsappendbenchmarkclf
forpenalty inl2 l1
print80
printspenalty  penaltyupper
  Train Liblinear model
resultsappendbenchmarkLinearSVCpenaltypenalty dualFalse
tol1e3
  Train SGD model
resultsappendbenchmarkSGDClassifieralpha0001 maxiter50
penaltypenalty
  Train SGD with Elastic Net penalty
print80
printElasticNet penalty
resultsappendbenchmarkSGDClassifieralpha0001 maxiter50
penaltyelasticnet
  Train NearestCentroid without threshold
print80
printNearestCentroid aka Rocchio classifier
resultsappendbenchmarkNearestCentroid
  Train sparse Naive Bayes classifiers
print80
printNaive Bayes
resultsappendbenchmarkMultinomialNBalpha01
resultsappendbenchmarkBernoulliNBalpha01
resultsappendbenchmarkComplementNBalpha1
print80
printLinearSVC with L1based feature selection
  The smaller C the stronger the regularization
  The more regularization the more sparsity
resultsappendbenchmarkPipeline
featureselection SelectFromModelLinearSVCpenaltyl1 dualFalse
tol1e3
classification LinearSVCpenaltyl2
  make some plots
indices  nparangelenresults
1446 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
results xi forxinresults foriinrange4
clfnames score trainingtime testtime results
trainingtime nparraytrainingtime npmaxtrainingtime
testtime nparraytesttime npmaxtesttime
pltfigurefigsize12 8
plttitleScore
pltbarhindices score 2 labelscore colornavy
pltbarhindices 3 trainingtime 2 labeltraining time
colorc
pltbarhindices 6 testtime 2 labeltest time colordarkorange
plticks
pltlegendlocbest
pltsubplotsadjustleft25
pltsubplotsadjusttop95
pltsubplotsadjustbottom05
fori cinzipindices clfnames
plttext3 i c
pltshow
Total running time of the script 0 minutes 4728 seconds
529 Decision Trees
Examples concerning the sklearntree module
Note Click here to download the full example code
5291 Decision Tree Regression
A 1D regression with decision tree
Thedecision trees is used to fit a sine curve with addition noisy observation As a result it learns local linear regres
sions approximating the sine curve
We can see that if the maximum depth of the tree controlled by the maxdepth parameter is set too high the
decision trees learn too fine details of the training data and learn from the noise ie they overfit
529 Decision Trees 1447
```

```
scikitlearn user guide Release 0213
printdoc
  Import the necessary modules and libraries
import numpy as np
from sklearn.tree import DecisionTreeRegressor
import matplotlib.pyplot as plt
  Create a random dataset
rng = np.random.RandomState(1)
X = np.sort(5 * rng.rand(80, 1), axis=0)
y = np.sin(X).ravel()
y[5:30] = 0.5 * rng.rand(16)
  Fit regression model
regr1 = DecisionTreeRegressor(max_depth=2)
regr2 = DecisionTreeRegressor(max_depth=5)
regr1.fit(X, y)
regr2.fit(X, y)
  Predict
Xtest = np.arange(0.0, 0.5, 0.01).reshape((100, 1))
y1 = regr1.predict(Xtest)
y2 = regr2.predict(Xtest)
  Plot the results
plt.figure
1448 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
pltscatterX y s20 edgecolorblack
cdarkorange labeldata
pltplotXtest y1 colorcornflowerblue
labelmaxdepth2 linewidth2
pltplotXtest y2 coloryellowgreen labelmaxdepth5 linewidth2
pltxlabeldata
pltylabeltarget
plttitleDecision Tree Regression
pltlegend
pltshow
```

Total running time of the script 0 minutes 0124 seconds

Note [Click here to download the full example code](#)

5292 Multioutput Decision Tree Regression

An example to illustrate multioutput regression with decision tree
Thedecision trees is used to predict simultaneously the noisy x and y observations of a circle given a single underlying
feature As a result it learns local linear regressions approximating the circle
We can see that if the maximum depth of the tree controlled by the maxdepth parameter is set too high the
decision trees learn too fine details of the training data and learn from the noise ie they overfit
529 Decision Trees 1449

```
scikitlearn user guide Release 0213
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeRegressor
Create a random dataset
rng = np.random.RandomState(1)
X = np.sort(200 * rng.rand(100, 1) - 100, axis=0)
y = np.array(np.pi * np.sin(X.ravel()) + rng.rand(100))
y5 = 0.5 * rng.rand(20)
Fit regression model
regr1 = DecisionTreeRegressor(max_depth=2)
regr2 = DecisionTreeRegressor(max_depth=5)
regr3 = DecisionTreeRegressor(max_depth=8)
regr1.fit(X, y)
regr2.fit(X, y)
regr3.fit(X, y)
Predict
Xtest = np.arange(-100, 100, 0.01).reshape((1000, 1))
y1 = regr1.predict(Xtest)
y2 = regr2.predict(Xtest)
y3 = regr3.predict(Xtest)
1450 Chapter 5 Examples
```


scikitlearn user guide Release 0213

Plot the results

```
pltfigure
s 25
pltscattery 0 y 1 cnavy ss
edgecolorblack labeldata
pltscattery1 0 y1 1 ccornflowerblue ss
edgecolorblack labelmaxdepth2
pltscattery2 0 y2 1 cred ss
edgecolorblack labelmaxdepth5
pltscattery3 0 y3 1 corange ss
edgecolorblack labelmaxdepth8
pltxlim6 6
pltylim6 6
pltxlabeltarget 1
pltylabeltarget 2
plttitleMultioutput Decision Tree Regression
pltlegendlocbest
pltshow
```

Total running time of the script 0 minutes 0109 seconds

Note [Click here to download the full example code](#)

5293 Plot the decision surface of a decision tree on the iris dataset

Plot the decision surface of a decision tree trained on pairs of features of the iris dataset

Seedecision tree for more information on the estimator

For each pair of iris features the decision tree learns decision boundaries made of combinations of simple thresholding rules inferred from the training samples

We also show the tree structure of a model built on all of the features

529 Decision Trees 1451

scikitlearn user guide Release 0213

```
•
printdoc
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import loadiris
from sklearn.tree import DecisionTreeClassifier plottree
Parameters
nclasses 3
plotcolors ryb
plotstep 002
Load data
iris loadiris
forpairidx pair in enumerate(0 1 0 2 0 3
1 2 1 3 2 3
We only take the two corresponding features
X irisdata pair
y iristarget
Train
clf DecisionTreeClassifierfitX y
Plot the decision boundary
pltsubplot2 3 pairidx 1
529 Decision Trees 1453
```

```
scikitlearn user guide Release 0213
xmin xmax X 0min 1 X 0max 1
ymin ymax X 1min 1 X 1max 1
xx yy npmeshgridnparangexmin xmax plotstep
nparangeymin ymax plotstep
plttightlAYOUTpad05 wpad05 pad25
Z clfpredictnpcxxravel yy.ravel
Z Zreshapexxshape
cs pltcontourfxx yy Z cmappltcmRdYIBu
pltxlabelirisfeaturenamespair0
pltylabelirisfeaturenamespair1
Plot the training points
fori color inziprangenclasses plotcolors
idx npwherey i
pltscatterXidx 0 Xidx 1 ccolor labeliristargetnamesi
cmappltcmRdYIBu edgecolorblack s15
pltstitleDecision surface of a decision tree using paired features
pltlegendloclower right borderpad0 handletextpad0
pltaxistight
pltfigure
clf DecisionTreeClassifierfitirisdata iristarget
plottreeclf filledTrue
pltshow
Total running time of the script 0 minutes 0934 seconds
Note Click here to download the full example code
5294 Understanding the decision tree structure
The decision tree structure can be analysed to gain further insight on the relation between the features and the target
to predict In this example we show how to retrieve
• the binary tree structure
• the depth of each node and whether or not it's a leaf
• the nodes that were reached by a sample using the decisionpath method
• the leaf that was reached by a sample using the apply method
• the rules that were used to predict a sample
• the decision path shared by a group of samples
Out
The binary tree structure has 5 nodes and has the following tree structure
node0 test node go to node 1 if X 3 0800000011920929 else to node 2
node1 leaf node
node2 test node go to node 3 if X 2 4950000047683716 else to node 4
node3 leaf node
1454 Chapter 5 Examples
```

scikitlearn user guide Release 0213

node4 leaf node

Rules used to predict sample 0

decision id node 0 Xtest0 3 24 0800000011920929

decision id node 2 Xtest0 2 51 4950000047683716

The following samples 0 1 share the node 0 2 in the tree

It is 400 of all nodes

import numpy as np

from sklearnmodelselection import traintestsplit

from sklearndatasets import loadiris

from sklearnntree import DecisionTreeClassifier

iris loadiris

X irisdata

y iristarget

Xtrain Xtest ytrain ytest traintestsplitX y randomstate0

estimator DecisionTreeClassifiermaxleafnodes3 randomstate0

estimatorfitXtrain ytrain

The decision estimator has an attribute called tree which stores the entire tree structure and allows access to low level attributes The binary tree is represented as a number of parallel arrays The ith element of each array holds information about the node i Node 0 is the trees root NOTE Some of the arrays only apply to either leaves or split nodes resp In this case the values of nodes of the other type are arbitrary

Among those arrays we have

leftchild id of the left child of the node

rightchild id of the right child of the node

feature feature used for splitting the node

threshold threshold value at the node

Using those arrays we can parse the tree structure

nnodes estimatortreenodecount

childrenleft estimatortreechildrenleft

childrenright estimatortreechildrenright

feature estimatortreefeature

threshold estimatortreethreshold

The tree structure can be traversed to compute various properties such

as the depth of each node and whether or not it is a leaf

nodedepth npzerosshapennodes dtypepint64

isleaves npzerosshapennodes dtypebool

stack 0 1 seed is the root node id and its parent depth

whilelenstack 0

nodeid parentdepth stackpop

529 Decision Trees 1455

```
scikitlearn user guide Release 0213
nodedepthnodeid parentdepth 1
  If we have a test node
ifchildrenleftnodeid childrenrightnodeid
stackappendchildrenleftnodeid parentdepth 1
stackappendchildrenrightnodeid parentdepth 1
else
isleavesnodeid True
printThe binary tree structure has snodes and has
the following tree structure
  nnodes
  foriinrangennodes
  ifisleavesi
  printsnodesleaf node nodedepthi t i
  else
  printsnodestest node go to node sif X s selse to
  nodes
    nodedepthi t
  i
  childrenlefti
  featurei
  thresholdi
  childrenrighti

print
  First lets retrieve the decision path of each sample The decisionpath
  method allows to retrieve the node indicator functions A non zero element of
  indicator matrix at the position i j indicates that the sample i goes
  through the node j
nodeindicator estimatordecisionpathXtest
  Similarly we can also have the leaves ids reached by each sample
leaveid estimatorapplyXtest
  Now its possible to get the tests that were used to predict a sample or
  a group of samples First lets make it for the sample
sampleid 0
nodeindex nodeindicatorindicesnodeindicatorindptrsampleid
nodeindicatorindptrsampleid 1
printRules used to predict sample s sampleid
  fornodeid innodeindex
  ifleaveidsampleid nodeid
  continue
  ifXtestsampleid featurenodeid thresholdnodeid
  thresholdsign
  else
  thresholdsign
  printdecision id node s Xtest ss ss s
  nodeid
  sampleid
1456 Chapter 5 Examples
```

```
scikitlearn user guide Release 0213
featurenodeid
Xtestsampleid featurenodeid
thresholdsign
thresholdnodeid
For a group of samples we have the following common node
sampleids 0 1
commonnodes nodeindicatortoarraysampleidssumaxis0
lensampleids
commonnodeid nparangenodescommonnodes
printnThe following samples sshare the node sin the tree
sampleids commonnodeid
printIt is s of all nodes 100 lencommonnodeid nnodes
Total running time of the script 0 minutes 0003 seconds
529 Decision Trees 1457
```


CHAPTER

SIX

API (Application Programming Interface) REFERENCE

This is the class and function reference of scikitlearn Please refer to the full user guide for further details as the class and function raw specifications may not be enough to give full guidelines on their uses For reference on concepts repeated across the API (Application Programming Interface) see Glossary of Common Terms and API (Application Programming Interface) Element

61sklearnbase Base classes and utility functions

Base classes for all estimators

611 Base classes

baseBaseEstimator Base class for all estimators in scikitlearn

baseBiclustermixin Mixin class for all biclustermixin estimators in scikitlearn

baseClassifierMixin Mixin class for all classifiers in scikitlearn

baseClusterMixin Mixin class for all cluster estimators in scikitlearn

baseDensityMixin Mixin class for all density estimators in scikitlearn

baseRegressorMixin Mixin class for all regression estimators in scikitlearn

baseTransformerMixin Mixin class for all transformers in scikitlearn

sklearnbase BaseEstimator

classsklearnbase BaseEstimator

Base class for all estimators in scikitlearn

Notes

All estimators should specify all the parameters that can be set at the class level in their init as explicit

keyword arguments no args orkwargs

Methods

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

1459

scikitlearn user guide Release 0213

init selfargs kwargs

Initialize self See helptypeself for accurate signature

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnbaseBaseEstimator

- Inductive Clustering

- Column Transformer with Heterogeneous Data Sources

sklearnbase Biclustermixin

classssklearnbase Biclustermixin

Mixin class for all bicluster estimators in scikitlearn

Attributes

biclusters Convenient way to get row and column indicators together

Methods

getindices self i Row and column indices of the i'th bicluster

getshape self i Shape of the i'th bicluster

getsubmatrix self i data Returns the submatrix corresponding to bicluster i

init selfargs kwargs

Initialize self See helptypeself for accurate signature

biclusters

Convenient way to get row and column indicators together

Returns the rows andcolumns members

getindices selfi

Row and column indices of the i'th bicluster

1460 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Only works if rows andcolumns attributes exist

Parameters

iint The index of the cluster

Returns

rowind nparray dtypepintp Indices of rows in the dataset that belong to the bicluster

colind nparray dtypepintp Indices of columns in the dataset that belong to the biclus

ter

getshape selfi

Shape of the i'th bicluster

Parameters

iint The index of the cluster

Returns

shape int int Number of rows and columns resp in the bicluster

getsubmatrix selfidata

Returns the submatrix corresponding to bicluster i

Parameters

iint The index of the cluster

data array The data

Returns

submatrix array The submatrix corresponding to bicluster i

Notes

Works with sparse matrices Only works if rows andcolumns attributes exist

sklearnbase ClassifierMixin

classsklearnbase ClassifierMixin

Mixin class for all classifiers in scikitlearn

Methods

score self X y sampleweight Returns the mean accuracy on the given test data and labels

init selfargs kwargs

Initialize self See helptypeself for accurate signature

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

61sklearnbase Base classes and utility functions 1461

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

sklearnbase ClusterMixin

classsklearnbase ClusterMixin

Mixin class for all cluster estimators in scikitlearn

Methods

fitpredict self X y Performs clustering on X and returns cluster labels

init selfargs kwargs

Initialize self See helptypeself for accurate signature

fitpredict selfXyNone

Performs clustering on X and returns cluster labels

Parameters

Xndarray shape nsamples nfeatures Input data

yIgnored not used present for API (Application Programming Interface) consistency by convention

Returns

labels ndarray shape nsamples cluster labels

sklearnbase DensityMixin

classsklearnbase DensityMixin

Mixin class for all density estimators in scikitlearn

Methods

score self X y Returns the score of the model on the data X

init selfargs kwargs

Initialize self See helptypeself for accurate signature

scoreselfXyNone

Returns the score of the model on the data X

Parameters

Xarraylike shape nsamples nfeatures

1462 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns
score float
sklearnbase RegressorMixin
class sklearnbase RegressorMixin
Mixin class for all regression estimators in scikitlearn
Methods
score self X y sampleweight Returns the coefficient of determination R2 of the prediction
init selfargs kwargs
Initialize self See helptypeself for accurate signature
score selfXsampleweightNone
Returns the coefficient of determination R2 of the prediction
The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$
2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score
is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always
predicts the expected value of y disregarding the input features would get a R2 score of 0.0
Parameters
Xarraylike shape (n_samples, n_features) Test samples For some estimators this may
be a precomputed kernel matrix instead shape (n_samples, n_samplesfitted) where
n_samplesfitted is the number of samples used in the fitting for the estimator
yarraylike shape (n_samples) or (n_samples, n_outputs) True values for X
sampleweight arraylike shape (n_samples) optional Sample weights
Returns
score float R2 of selfpredictX wrt y
Notes
The R2 score used when calling score on a regressor will use multioutputuniformaverage
from version 0.23 to keep consistent with metricsr2score This will influence the score
method of all the multioutput regressors except for multioutputMultiOutputRegressor
To specify the default value manually and avoid the warning please either call metricsr2score
directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses
multioutputuniformaverage
sklearnbase TransformerMixin
class sklearnbase TransformerMixin
Mixin class for all transformers in scikitlearn
61sklearnbase Base classes and utility functions 1463

scikitlearn user guide Release 0213

Methods

fittransform self X y Fit to data then transform it

init selfargs kwargs

Initialize self See helptypeself for accurate signature

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

Examples using sklearnbaseTransformerMixin

•Column Transformer with Heterogeneous Data Sources

612 Functions

baseclone estimator safe Constructs a new estimator with the same parameters

baseisclassifier estimator Returns True if the given estimator is probably a classifier

baseisregressor estimator Returns True if the given estimator is probably a regressor

configcontext newconfig Context manager for global scikitlearn configuration

getconfig Retrieve current values for configuration set by

setconfig

setconfig assumefinite workingmemory Set global scikitlearn configuration

showversions Print useful debugging information

sklearnbase clone

sklearnbase cloneestimator safeTrue

Constructs a new estimator with the same parameters

Clone does a deep copy of the model in an estimator without actually copying attached data It yields a new estimator with the same parameters that has not been fit on any data

Parameters

estimator estimator object or list tuple or set of objects The estimator or group of estimators to be cloned

safe boolean optional If safe is false clone will fall back to a deep copy on objects that are not estimators

1464 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearnbase isclassifier

sklearnbase isclassifier estimator

Returns True if the given estimator is probably a classifier

Parameters

estimator object Estimator object to test

Returns

out bool True if estimator is a classifier and False otherwise

sklearnbase isregressor

sklearnbase isregressor estimator

Returns True if the given estimator is probably a regressor

Parameters

estimator object Estimator object to test

Returns

out bool True if estimator is a regressor and False otherwise

sklearn configcontext

sklearn configcontext newconfig

Context manager for global scikitlearn configuration

Parameters

assumeinfinite bool optional If True validation for finiteness will be skipped saving time but

leading to potential crashes If False validation for finiteness will be performed avoiding

error Global default False

workingmemory int optional If set scikitlearn will attempt to limit the size of temporary

arrays to this number of MiB per job when parallelised often saving both computation

time and memory on expensive operations that can be performed in chunks Global default

1024

See also

setconfig Set global scikitlearn configuration

getconfig Retrieve current values of the global configuration

Notes

All settings not just those presently modified will be returned to their previous values when the context manager

is exited This is not threadsafe

Examples

61sklearnbase Base classes and utility functions 1465

scikitlearn user guide Release 0213

```
import sklearn
from sklearnutilsvalidation import assertallfinite
with sklearnconfigcontextassumeefinite True
assertallfinitefloatnan
with sklearnconfigcontextassumeefinite True
with sklearnconfigcontextassumeefinite False
assertallfinitefloatnan
```

Traceback most recent call last

ValueError Input contains NaN
sklearn getconfig
sklearn getconfig
Retrieve current values for configuration set by setconfig
Returns
config dict Keys are parameter names that can be passed to setconfig
See also
configcontext Context manager for global scikitlearn configuration
setconfig Set global scikitlearn configuration
sklearn setconfig
sklearn setconfig assumeefiniteNone workingmemoryNone printchangedonlyNone
Set global scikitlearn configuration
New in version 019
Parameters
assumeefinite bool optional If True validation for finiteness will be skipped saving time but
leading to potential crashes If False validation for finiteness will be performed avoiding
error Global default False
New in version 019
workingmemory int optional If set scikitlearn will attempt to limit the size of temporary
arrays to this number of MiB per job when parallelised often saving both computation
time and memory on expensive operations that can be performed in chunks Global default
1024
New in version 020
printchangedonly bool optional If True only the parameters that were set to nondefault
values will be printed when printing an estimator For example printSVC while
True will only print 'SVC (Support Vector Classification)' while the default behaviour would be to print 'SVCC10
cachesize200 ' with all the nonchanged parameters
New in version 021
See also
configcontext Context manager for global scikitlearn configuration
1466 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getConfig Retrieve current values of the global configuration

Examples using sklearnsetconfig

•Compact estimator representations

sklearn showversions

sklearn showversions

Print useful debugging information

62sklearncalibration Probability Calibration

Calibration of predicted probabilities

User guide See the Probability calibration section for further details

calibrationCalibratedClassifierCV Probability calibration with isotonic regression or sigmoid

621sklearncalibration CalibratedClassifierCV

classsklearncalibration CalibratedClassifierCV baseestimatorNone

method'sigmoid' cv'warn'

Probability calibration with isotonic regression or sigmoid

See glossary entry for crossvalidation estimator

With this class the baseestimator is fit on the train set of the crossvalidation generator and the test set is used

for calibration The probabilities for each of the folds are then averaged for prediction In case that cv"prefit"

is passed to init it is assumed that baseestimator has been fitted already and all data is used for calibration

Note that data for fitting the classifier and for calibrating it must be disjoint

Read more in the User Guide

Parameters

baseestimator instance BaseEstimator The classifier whose output decision function needs

to be calibrated to offer more accurate predictproba outputs If cvprefit the classifier must

have been fit already on data

method 'sigmoid' or 'isotonic' The method to use for calibration Can be 'sigmoid' which

corresponds to Platt's method or 'isotonic' which is a nonparametric approach It is not

advised to use isotonic calibration with too few calibration samples 1000 since it

tends to overfit Use sigmoids Platt's calibration in this case

cvinteger crossvalidation generator iterable or "prefit" optional Determines the cross

validation splitting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation
- integer to specify the number of folds
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

62sklearncalibration Probability Calibration 1467

scikitlearn user guide Release 0213

For integerNone inputs if yis binary or multiclass sklearnmodelselection
StratifiedKFold is used If yis neither binary nor multiclass sklearn
modelselectionKFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

If “prefit” is passed it is assumed that baseestimator has been fitted already and all data is
used for calibration

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in
v022

Attributes

classes array shape nclasses The class labels

calibratedclassifiers list len equal to cv or 1 if cv “prefit” The list of calibrated
classifiers one for each crossvalidation fold which has been fitted on all but the validation
fold and calibrated on the validation fold

References

R57cf438d70601 R57cf438d70602 R57cf438d70603 R57cf438d70604

Methods

fitself X y sampleweight Fit the calibrated model

getparams self deep Get parameters for this estimator

predict self X Predict the target of new samples

predictproba self X Posterior probabilities of classification

score self X y sampleweight Returns the mean accuracy on the given test data and
labels

setparams self params Set the parameters of this estimator

init selfbaseestimatorNone method’sigmoid’ cv’warn’

fitselfXysampleweightNone

Fit the calibrated model

Parameters

Xarraylike shape nsamples nfeatures Training data

yarraylike shape nsamples Target values

sampleweight arraylike shape nsamples or None Sample weights If None then
samples are equally weighted

Returns

self object Returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

1468 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict the target of new samples Can be different from the prediction of the uncalibrated classifier

Parameters

Xarraylike shape nsamples nfeatures The samples

Returns

Carray shape nsamples The predicted class

predictproba selfX

Posterior probabilities of classification

This function returns posterior probabilities of classification according to each class on an array of test vectors X

Parameters

Xarraylike shape nsamples nfeatures The samples

Returns

Carray shape nsamples nclasses The predicted probas

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearncalibrationCalibratedClassifierCV

- Probability Calibration curves
 - Probability calibration of classifiers
 - Probability Calibration for 3class classification
- 62sklearncalibration Probability Calibration 1469

scikitlearn user guide Release 0213

calibrationcalibrationcurve ytrue

yprobCompute true and predicted probabilities for a calibration curve

622sklearncalibration calibrationcurve

sklearncalibration calibrationcurve ytrue yprob normalizeFalse nbins5 strategy'uniform'

Compute true and predicted probabilities for a calibration curve

The method assumes the inputs come from a binary classifier

Calibration curves may also be referred to as reliability diagrams

Read more in the User Guide

Parameters

ytrue array shape nsamples True targets

yprob array shape nsamples Probabilities of the positive class

normalize bool optional defaultFalse Whether yprob needs to be normalized into the bin

0 1 ie is not a proper probability If True the smallest value in yprob is mapped onto

0 and the largest one onto 1

nbins int Number of bins A bigger number requires more data Bins with no data points

ie without corresponding values in yprob will not be returned thus there may be fewer

than nbins in the return value

strategy 'uniform' 'quantile' default'uniform' Strategy used to define the widths of the bins

uniform All bins have identical widths

quantile All bins have the same number of points

Returns

probtrue array shape nbins or smaller The true probability in each bin fraction of positives

probpred array shape nbins or smaller The mean predicted probability in each bin

References

Alexandru NiculescuMizil and Rich Caruana 2005 Predicting Good Probabilities With Supervised Learning in Proceedings of the 22nd International Conference on Machine Learning ICML See section 4 Qualitative

Analysis of Predictions

Examples using sklearncalibrationcalibrationcurve

•Comparison of Calibration of Classifiers

•Probability Calibration curves

1470 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

63sklearncluster Clustering

Thesklearncluster module gathers popular unsupervised clustering algorithms

User guide See the Clustering section for further details

631 Classes

clusterAffinityPropagation damping Perform Affinity Propagation Clustering of data

clusterAgglomerativeClustering Agglomerative Clustering

clusterBirch threshold branchingfactor Implements the Birch clustering algorithm

clusterDBSCAN eps minsamples metric Perform DBSCAN clustering from vector array or distance matrix

clusterOPTICS minsamples maxeps Estimate clustering structure from vector array

clusterFeatureAgglomeration nclusters

Agglomerate features

clusterKMeans nclusters init ninit KMeans clustering

clusterMiniBatchKMeans nclusters init MiniBatch KMeans clustering

clusterMeanShift bandwidth seeds Mean shift clustering using a flat kernel

clusterSpectralClustering nclusters Apply clustering to a projection of the normalized Laplacian

sklearncluster AffinityPropagation

classsklearncluster AffinityPropagation damping05 maxiter200 conver

genceiter15 copyTrue preferenceNone

affinity'euclidean' verboseFalse

Perform Affinity Propagation Clustering of data

Read more in the User Guide

Parameters

damping float optional default 05 Damping factor between 05 and 1 is the extent to which the current value is maintained relative to incoming values weighted 1 damping

This in order to avoid numerical oscillations when updating these values messages

maxiter int optional default 200 Maximum number of iterations

convergenceiter int optional default 15 Number of iterations with no change in the number of estimated clusters that stops the convergence

copy boolean optional default True Make a copy of input data

preference arraylike shape nsamples or float optional Preferences for each point points with larger values of preferences are more likely to be chosen as exemplars The number of exemplars ie of clusters is influenced by the input preferences value If the preferences are not passed as arguments they will be set to the median of the input similarities

affinity string optional default''euclidean'' Which affinity to use At the moment precomputed andeuclidean are supported euclidean uses the negative squared

euclidean distance between points

verbose boolean optional default False Whether to be verbose

Attributes

63sklearncluster Clustering 1471

scikitlearn user guide Release 0213

clustercentersindices array shape nclusters Indices of cluster centers
clustercenters array shape nclusters nfeatures Cluster centers if affinity
precomputed

labels array shape nsamples Labels of each point
affinitymatrix array shape nsamples nsamples Stores the affinity matrix used in fit
niter int Number of iterations taken to converge

Notes

For an example see examplesclusterplotaffinitypropagationpy
The algorithmic complexity of affinity propagation is quadratic in the number of points
Whenfit does not converge clustercenters becomes an empty array and all training samples will be
labelled as1 In addition predict will then label every sample as 1
When all training samples have equal similarities and equal preferences the assignment of cluster centers and
labels depends on the preference If the preference is smaller than the similarities fit will result in a single
cluster center and label 0for every sample Otherwise every training sample becomes its own cluster center
and is assigned a unique label

References

Brendan J Frey and Delbert Dueck “Clustering by Passing Messages Between Data Points” Science Feb 2007

Examples

```
from sklearncluster import AffinityPropagation
import numpy as np
X = np.array([2 1 4 1 0
              4 2 4 4 0])
clustering = AffinityPropagationfit(X)
clustering
AffinityPropagationaffinityeuclidean convergenceiter15 copyTrue
damping05 maxiter200 preferenceNone verboseFalse
clusteringlabels
array([0 0 1 1 1])
clusteringpredict([0 0 4 4])
array([0 1])
clusteringclustercenters
array([1 2
       4 2])
```

Methods

fitself X y Create affinity matrix from negative euclidean distances then apply affinity propagation clustering
fitpredict self X y Performs clustering on X and returns cluster labels
Continued on next page
1472 Chapter 6 API (Application Programming Interface) Reference

getparams self deep Get parameters for this estimator
predict self X Predict the closest cluster each sample in X belongs to
setparams self params Set the parameters of this estimator
init selfdamping05 maxiter200 convergenceiter15 copyTrue preferenceNone
affinity'euclidean' verboseFalse
fitselfXyNone

Create affinity matrix from negative euclidean distances then apply affinity propagation clustering

Parameters
Xarraylike shape nsamples nfeatures or nsamples nsamples Data matrix or if
affinity isprecomputed matrix of similarities affinities
yIgnored
fitpredict selfXyNone

Performs clustering on X and returns cluster labels

Parameters
Xndarray shape nsamples nfeatures Input data
yIgnored not used present for API (Application Programming Interface) consistency by convention
Returns

labels ndarray shape nsamples cluster labels

getparams selfdeepTrue
Get parameters for this estimator

Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns
params mapping of string to any Parameter names mapped to their values

predictselfX
Predict the closest cluster each sample in X belongs to

Parameters
Xarraylike sparse matrix shape nsamples nfeatures New data to predict
Returns

labels array shape nsamples Index of the cluster each sample belongs to
setparams selfparams

Set the parameters of this estimator
The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it's possible to update each component
of a nested object

Returns
self

scikitlearn user guide Release 0213

Examples using sklearnclusterAffinityPropagation

- Demo of affinity propagation clustering algorithm
- Comparing different clustering algorithms on toy datasets

sklearncluster AgglomerativeClustering

classsklearncluster AgglomerativeClustering nclusters2 affinity'euclidean' mem

oryNone connectivityNone com

putefulltree'auto' linkage'ward'

poolingfunc'deprecated' dis

tancethresholdNone

Agglomerative Clustering

Recursively merges the pair of clusters that minimally increases a given linkage distance

Read more in the User Guide

Parameters

nclusters int or None optional default2 The number of clusters to find It must be None

ifdistancethreshold is notNone

affinity string or callable default "euclidean" Metric used to compute the linkage Can be

"euclidean" "l1" "l2" "manhattan" "cosine" or "precomputed" If linkage is "ward"

only "euclidean" is accepted If "precomputed" a distance matrix instead of a similarity

matrix is needed as input for the fit method

memory None str or object with the joblibMemory interface optional Used to cache the

output of the computation of the tree By default no caching is done If a string is given it

is the path to the caching directory

connectivity arraylike or callable optional Connectivity matrix Defines for each sample the

neighboring samples following a given structure of the data This can be a connectivity

matrix itself or a callable that transforms the data into a connectivity matrix such as de

rived from kneighborsgraph Default is None ie the hierarchical clustering algorithm is

unstructured

computealltree bool or 'auto' optional Stop early the construction of the tree at

nclusters This is useful to decrease computation time if the number of clusters is not small

compared to the number of samples This option is useful only when specifying a connec

tivity matrix Note also that when varying the number of clusters and using caching it may

be advantageous to compute the full tree It must be True ifdistancethreshold is

notNone

linkage "ward" "complete" "average" "single" optional default"ward" Which linkage

criterion to use The linkage criterion determines which distance to use between sets of

observation The algorithm will merge the pairs of cluster that minimize this criterion

- ward minimizes the variance of the clusters being merged
- average uses the average of the distances of each observation of the two sets
- complete or maximum linkage uses the maximum distances between all observations of the two sets

- single uses the minimum of the distances between all observations of the two sets

1474 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

poolingfunc callable default'deprecated' Ignored

Deprecated since version 020 poolingfunc has been deprecated in 020 and will be removed in 022

distancethreshold float optional defaultNone The linkage distance threshold above which clusters will not be merged If not None nclusters must beNone and computefulltree must beTrue

New in version 021

Attributes

nclusters int The number of clusters found by the algorithm If

distancethresholdNone it will be equal to the given nclusters

labels array nsamples cluster labels for each point

nleaves int Number of leaves in the hierarchical tree

nconnectedcomponents int The estimated number of connected components in the graph

children arraylike shape nsamples1 2 The children of each nonleaf node Values less than nsamples correspond to leaves of the tree which are the original samples A node igreater than or equal to nsamples is a nonleaf node and has children childreni

nsamples Alternatively at the ith iteration childreni0 and childreni1 are merged to form node nsamples i

Examples

```
from sklearncluster import AgglomerativeClustering
```

```
import numpy as np
```

```
X = nparray1 2 1 4 1 0
```

```
4 2 4 4 4 0
```

```
clustering = AgglomerativeClusteringfitX
```

```
clustering
```

```
AgglomerativeClusteringaffinityeuclidean computefulltreeauto
```

```
connectivityNone distancethresholdNone
```

```
linkageward memoryNone nclusters2
```

```
poolingfuncdeprecated
```

```
clusteringlabels
```

```
array1 1 1 0 0 0
```

Methods

fitself X y Fit the hierarchical clustering on the data

fitpredict self X y Performs clustering on X and returns cluster labels

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

init self nclusters2 affinity'euclidean' memoryNone connectivityNone

computefulltree'auto' linkage'ward' poolingfunc'deprecated' dis

tancethresholdNone

fitselfXyNone

Fit the hierarchical clustering on the data

63sklearncluster Clustering 1475

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures Training data Shape nsamples
nfeatures or nsamples nsamples if affinity'precomputed'

ylgnored

Returns

self

fitpredict selfXyNone

Performs clustering on X and returns cluster labels

Parameters

Xndarray shape nsamples nfeatures Input data

ylgnored not used present for API (Application Programming Interface) consistency by convention

Returns

labels ndarray shape nsamples cluster labels

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnclusterAgglomerativeClustering

- Agglomerative clustering with and without structure
 - Various Agglomerative Clustering on a 2D embedding of digits
 - A demo of structured Ward hierarchical clustering on an image of coins
 - Hierarchical clustering structured vs unstructured ward
 - Agglomerative clustering with different metrics
 - Inductive Clustering
 - Comparing different hierarchical linkage methods on toy datasets
 - Comparing different clustering algorithms on toy datasets
- 1476 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearncluster Birch

class sklearncluster Birchthreshold05 branchingfactor50 nclusters3 com

putelabelsTrue copyTrue

Implements the Birch clustering algorithm

It is a memoryefficient onlinelearning algorithm provided as an alternative to MiniBatchKMeans It constructs a tree data structure with the cluster centroids being read off the leaf These can be either the final cluster centroids or can be provided as input to another clustering algorithm such as AgglomerativeClustering

Read more in the User Guide

Parameters

threshold float default 05 The radius of the subcluster obtained by merging a new sample and the closest subcluster should be lesser than the threshold Otherwise a new subcluster is started Setting this value to be very low promotes splitting and viceversa

branchingfactor int default 50 Maximum number of CF subclusters in each node If a new samples enters such that the number of subclusters exceed the branchingfactor then that node is split into two nodes with the subclusters redistributed in each The parent subcluster of that node is removed and two new subclusters are added as parents of the 2 split nodes nclusters int instance of sklearncluster model default 3 Number of clusters after the final clustering step which treats the subclusters from the leaves as new samples

•None the final clustering step is not performed and the subclusters are returned as they are

•sklearncluster Estimator If a model is provided the model is fit treating the subclusters as new samples and the initial data is mapped to the label of the closest subcluster

•int the model fit is AgglomerativeClustering withnclusters set to be equal to the int

computelabels bool default True Whether or not to compute labels for each fit

copy bool default True Whether or not to make a copy of the given data If set to False the initial data will be overwritten

Attributes

root CFNode Root of the CFTree

dummyleaf CFNode Start pointer to all the leaves

subclustercenters ndarray Centroids of all subclusters read directly from the leaves

subclusterlabels ndarray Labels assigned to the centroids of the subclusters after they are clustered globally

labels ndarray shape nsamples Array of labels assigned to the input data if partialfit is used instead of fit they are assigned to the last batch of data

Notes

The tree data structure consists of nodes with each node consisting of a number of subclusters The maximum number of subclusters in a node is determined by the branching factor Each subcluster maintains a linear sum squared sum and the number of samples in that subcluster In addition each subcluster can also have a node as its child if the subcluster is not a member of a leaf node

63sklearncluster Clustering 1477

scikitlearn user guide Release 0213

For a new point entering the root it is merged with the subcluster closest to it and the linear sum squared sum and the number of samples of that subcluster are updated This is done recursively till the properties of the leaf node are updated

References

- Tian Zhang Raghu Ramakrishnan Maron Livny BIRCH An efficient data clustering method for large databases <https://www.cs.fu.ca/Course/Central459/hanpapers/zhang96.pdf>
- Roberto Perdisci JBirch Java implementation of BIRCH clustering algorithm <https://code.google.com/archive/p/jbirch/>

Examples

```
from sklearn.cluster import Birch
X = [[0, 1, 0.3], [1, 0.3, 1], [0, 1, 0.3], [1, 0.3, 1]]
brc = Birch(branching_factor=50, n_clusters=None, threshold=0.5,
             compute_labels=True)
brc.fit(X)
Birch(branching_factor=50, compute_labels=True, copy=True, n_clusters=None,
       threshold=0.5)
brc.predict(X)
array([0, 0, 1, 1])
```

Methods

- fitself X y Build a CF Tree for the input data
- fitpredict self X y Performs clustering on X and returns cluster labels
- fittransform self X y Fit to data then transform it
- getparams self deep Get parameters for this estimator
- partialfit self X y Online learning
- predict self X Predict data using the centroids of subclusters
- setparams self params Set the parameters of this estimator
- transform self X Transform X into subcluster centroids dimension
- init selfthreshold=0.5 branching_factor=50 n_clusters=3 compute_labels=True

copy=True

fitselfXyNone

Build a CF Tree for the input data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Input data

ylgnored

fitpredict selfXyNone

Performs clustering on X and returns cluster labels

Parameters

Xndarray shape nsamples nfeatures Input data

scikitlearn user guide Release 0213

ylgnored not used present for API (Application Programming Interface) consistency by convention

Returns

labels ndarray shape nsamples cluster labels

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXNone yNone

Online learning Prevents rebuilding of CFTree from scratch

Parameters

Xarraylike sparse matrix shape nsamples nfeatures None Input data If X is not

provided only the global clustering step is done

ylgnored

predictselfX

Predict data using the centroids of subclusters

Avoid computation of the row norms of X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Input data

Returns

labels ndarray shapensamples Labelled data

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

63sklearncluster Clustering 1479

scikitlearn user guide Release 0213

transform selfX

Transform X into subcluster centroids dimension

Each dimension represents the distance from the sample point to each cluster centroid

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Input data

Returns

Xtrans arraylike sparse matrix shape nsamples nclusters Transformed data

Examples using sklearnclusterBirch

- Compare BIRCH and MiniBatchKMeans
- Comparing different clustering algorithms on toy datasets

sklearncluster DBSCAN

classsklearncluster DBSCANeps05 minsamples5 metric'euclidean' metricparamsNone

algorithm'auto' leafsize30 pNone njobsNone

Perform DBSCAN clustering from vector array or distance matrix

DBSCAN DensityBased Spatial Clustering of Applications with Noise Finds core samples of high density

and expands clusters from them Good for data which contains clusters of similar density

Read more in the User Guide

Parameters

eps float optional The maximum distance between two samples for one to be considered as in the neighborhood of the other This is not a maximum bound on the distances of points within a cluster This is the most important DBSCAN parameter to choose appropriately for your data set and distance function

minsamples int optional The number of samples or total weight in a neighborhood for a point to be considered as a core point This includes the point itself

metric string or callable The metric to use when calculating distance between instances in a feature array If metric is a string or callable it must be one of the options allowed

by sklearnmetricspairwiselocalities for its metric parameter If metric

is "precomputed" X is assumed to be a distance matrix and must be square X may be

a sparse matrix in which case only "nonzero" elements may be considered neighbors for DBSCAN

New in version 017 metric precomputed to accept precomputed sparse matrix

metricparams dict optional Additional keyword arguments for the metric function

New in version 019

algorithm 'auto' 'balltree' 'kdtree' 'brute' optional The algorithm to be used by the NearestNeighbors module to compute pointwise distances and find nearest neighbors See NearestNeighbors module documentation for details

leafsize int optional default 30 Leaf size passed to BallTree or cKDTree This can affect the speed of the construction and query as well as the memory required to store the tree The optimal value depends on the nature of the problem

1480 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

pfloat optional The power of the Minkowski metric to be used to calculate distance between points

njobs int or None optional defaultNone The number of parallel jobs to run None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

Attributes

coresampleindices array shape ncoresamples Indices of core samples

components array shape ncoresamples nfeatures Copy of each core sample found by training

labels array shape nsamples Cluster labels for each point in the dataset given to fit

Noisy samples are given the label 1

See also

OPTICS A similar clustering at multiple values of eps Our implementation is optimized for memory usage

Notes

For an example see examplesclusterplotdbscanpy

This implementation bulkcomputes all neighborhood queries which increases the memory complexity to $O(dn)$ where d is the average number of neighbors while original DBSCAN had memory complexity $O(n)$ On It may attract a higher memory complexity when querying these nearest neighborhoods depending on the algorithm

One way to avoid the query complexity is to precompute sparse neighborhoods in chunks usingNearestNeighborsradiusneighborsgraph withmodedistance then using metricprecomputed here

Another way to reduce memory and computation time is to remove nearduplicate points and use sampleweight instead

clusterOPTICS provides a similar clustering with lower memory usage

References

Ester M H P Kriegel J Sander and X Xu “A DensityBased Algorithm for Discovering Clusters in Large Spatial Databases with Noise” In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining Portland OR AAAI Press pp 226231 1996

Schubert E Sander J Ester M Kriegel H P Xu X 2017 DBSCAN revisited revisited why and how you should still use DBSCAN ACM Transactions on Database Systems TODS 423 19

Examples

```
from sklearncluster import DBSCAN
import numpy as np
X = nparray1 2 2 2 2 3
8 7 8 8 25 80
clustering = DBSCANeps3 minsamples2fitX
clusteringlabels
63sklearncluster Clustering 1481
```

scikitlearn user guide Release 0213

array 0 0 0 1 1 1

clustering

DBSCANAlgorithmauto eps3 leafsize30 metriceuclidean

metricparamsNone minsamples2 njobsNone pNone

Methods

fitself X y sampleweight Perform DBSCAN clustering from features or distance matrix

fitpredict self X y sampleweight Performs clustering on X and returns cluster labels

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

init selfeps05 minsamples5 metric'euclidean' metricparamsNone algo

rithm'auto' leafsize30 pNone njobsNone

fitselfXyNone sampleweightNone

Perform DBSCAN clustering from features or distance matrix

Parameters

Xarray or sparse CSR (Compressed Sparse Row) matrix of shape nsamples nfeatures or array of shape nsamples nsamples A feature array or array of distances between samples if

metricprecomputed

sampleweight array shape nsamples optional Weight of each sample such that a sample with a weight of at least minsamples is by itself a core sample a sample with negative weight may inhibit its epsneighbor from being core Note that weights are absolute and default to 1

ylgnored

fitpredict selfXyNone sampleweightNone

Performs clustering on X and returns cluster labels

Parameters

Xarray or sparse CSR (Compressed Sparse Row) matrix of shape nsamples nfeatures or array of shape nsamples nsamples A feature array or array of distances between samples if

metricprecomputed

sampleweight array shape nsamples optional Weight of each sample such that a sample with a weight of at least minsamples is by itself a core sample a sample with negative weight may inhibit its epsneighbor from being core Note that weights are absolute and default to 1

ylgnored

Returns

yndarray shape nsamples cluster labels

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

1482 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnclusterDBSCAN

- Demo of DBSCAN clustering algorithm

- Comparing different clustering algorithms on toy datasets

sklearncluster OPTICS

classsklearncluster OPTICSminsamples5 maxepsinf metric'minkowski' p2met

ricparamsNone clustermethod'xi' epsNone xi005 prede

cessorcorrectionTrue minclustersizeNone algorithm'auto'

leafsize30 njobsNone

Estimate clustering structure from vector array

OPTICS Ordering Points To Identify the Clustering Structure closely related to DBSCAN finds core sample

of high density and expands clusters from them R2c55e37003fe1 Unlike DBSCAN keeps cluster hierarchy

for a variable neighborhood radius Better suited for usage on large datasets than the current sklearn implemen

tation of DBSCAN

Clusters are then extracted using a DBSCANlike method clustermethod 'dbscan' or an automatic technique

proposed in R2c55e37003fe1 clustermethod 'xi'

This implementation deviates from the original OPTICS by first performing knearestneighborhood searches

on all points to identify core sizes then computing only the distances to unprocessed points when constructing

the cluster order Note that we do not employ a heap to manage the expansion candidates so the time complexity

will be On2

Read more in the User Guide

Parameters

minsamples int 1 or float between 0 and 1 default5 The number of samples in a neigh

borhood for a point to be considered as a core point Also up and down steep regions can't

have more then minsamples consecutive nonsteep points Expressed as an absolute

number or a fraction of the number of samples rounded to be at least 2

maxeps float optional defaultnpinf The maximum distance between two samples for one

to be considered as in the neighborhood of the other Default value of npinf will identify

clusters across all scales reducing maxeps will result in shorter run times

metric string or callable optional default'minkowski' Metric to use for distance computa

tion Any metric from scikitlearn or scipyspatialdistance can be used

If metric is a callable function it is called on each pair of instances rows and the resulting

value recorded The callable should take two arrays as input and return one value indicating

63sklearncluster Clustering 1483

scikitlearn user guide Release 0213

the distance between them This works for Scipy’s metrics but is less efficient than passing the metric name as a string If metric is “precomputed” X is assumed to be a distance matrix and must be square

Valid values for metric are

- from scikitlearn ‘cityblock’ ‘cosine’ ‘euclidean’ ‘l1’ ‘l2’ ‘manhattan’
- from scipyspatialdistance ‘braycurtis’ ‘canberra’ ‘chebyshev’ ‘correlation’ ‘dice’ ‘hamming’ ‘jaccard’ ‘kulsinski’ ‘mahalanobis’ ‘minkowski’ ‘rogerstanimoto’ ‘rus sellrao’ ‘seuclidean’ ‘sokalmichener’ ‘sokalsneath’ ‘sqeuclidean’ ‘yule’

See the documentation for scipyspatialdistance for details on these metrics

pinteger optional default2 Parameter for the Minkowski metric from sklearn

metricspairwisel1distances When p 1 this is equivalent to using manhat

tandistance l1 and euclidean l2 for p 2 For arbitrary p minkowskidistance

lp is used

metricparams dict optional defaultNone Additional keyword arguments for the metric

function

clustermethod string optional default’xi’ The extraction method used to extract clusters

using the calculated reachability and ordering Possible values are “xi” and “dbscan”

eps float optional defaultNone The maximum distance between two samples for one to be

considered as in the neighborhood of the other By default it assumes the same value as

maxeps Used only when clustermethoddbscan

xifloat between 0 and 1 optional default005 Determines the minimum steepness on the

reachability plot that constitutes a cluster boundary For example an upwards point in the

reachability plot is defined by the ratio from one point to its successor being at most 1xi

Used only when clustermethodxi

predecessorcorrection bool optional defaultTrue Correct clusters according to the pre

decessors calculated by OPTICS R2c55e37003fe2 This parameter has minimal effect on

most datasets Used only when clustermethodxi

minclustersize int 1 or float between 0 and 1 defaultNone Minimum number of sam

ples in an OPTICS cluster expressed as an absolute number or a fraction of the number of

samples rounded to be at least 2 If None the value of minsamples is used instead

Used only when clustermethodxi

algorithm ‘auto’ ‘balltree’ ‘kdtree’ ‘brute’ optional Algorithm used to compute the

nearest neighbors

- ‘balltree’ will use BallTree
- ‘kdtree’ will use KDTree
- ‘brute’ will use a brute force search
- ‘auto’ will attempt to decide the most appropriate algorithm based on the values passed

tofit method default

Note fitting on sparse input will override the setting of this parameter using brute force

leafsize int optional default30 Leaf size passed to BallTree orKDTree This can

affect the speed of the construction and query as well as the memory required to store the

tree The optimal value depends on the nature of the problem

1484 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

njobs int or None optional defaultNone The number of parallel jobs to run for neighbors searchNone means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

Attributes

labels array shape nsamples Cluster labels for each point in the dataset given to fit Noisy samples and points which are not included in a leaf cluster of clusterhierarchy are labeled as 1

reachability array shape nsamples Reachability distances per sample indexed by object order Useclustreachabilityclustordering to access in cluster order

ordering array shape nsamples The cluster ordered list of sample indices

coredistances array shape nsamples Distance at which each sample becomes a core point indexed by object order Points which will never be core have a distance of inf Use clustcoredistancesclustordering to access in cluster order

predecessor array shape nsamples Point that a sample was reached from indexed by object order Seed points have a predecessor of 1

clusterhierarchy array shape nclusters 2 The list of clusters in the form of start end in each row with all indices inclusive The clusters are ordered according toend start ascending so that larger clusters encompassing smaller clusters come after those smaller ones Since labels does not reflect the hierarchy usually lenclusterhierarchy npuniqueopticslabels Please also note that these indices are of the ordering ieXorderingstartend 1form a cluster Only available when clustermethodxi

See also

DBSCAN A similar clustering for a specified neighborhood radius eps Our implementation is optimized for runtime

References

R2c55e37003fe1 R2c55e37003fe2

Methods

fitself X y Perform OPTICS clustering

fitpredict self X y Performs clustering on X and returns cluster labels

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

init selfminsamples5 maxepsinf metric'minkowski' p2metricparamsNone

clustermethod'xi' epsNone xi005 predecessorcorrectionTrue

minclustersizeNone algorithm'auto' leafsize30 njobsNone

fitselfXyNone

Perform OPTICS clustering

Extracts an ordered list of points and reachability distances and performs initial clustering using maxeps distance specified at OPTICS object instantiation

Parameters

63sklearncluster Clustering 1485

scikitlearn user guide Release 0213

Xarray shape nsamples nfeatures or nsamples nsamples if met
ric'precomputed' A feature array or array of distances between samples if met
ric'precomputed'

yignored

Returns

self instance of OPTICS The instance

fitpredict selfXyNone

Performs clustering on X and returns cluster labels

Parameters

Xndarray shape nsamples nfeatures Input data

yignored not used present for API (Application Programming Interface) consistency by convention

Returns

labels ndarray shape nsamples cluster labels

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnclusterOPTICS

•Demo of OPTICS clustering algorithm

•Comparing different clustering algorithms on toy datasets

sklearncluster FeatureAgglomeration

classsklearncluster FeatureAgglomeration nclusters2 affinity'euclidean' mem

oryNone connectivityNone com

putefulltree'auto' linkage'ward'

poolingfuncfunction mean dis

tancethresholdNone

Agglomerate features

Similar to AgglomerativeClustering but recursively merges features instead of samples

1486 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

nclusters int or None optional default2 The number of clusters to find It must be None

ifdistancethreshold is notNone

affinity string or callable default “euclidean” Metric used to compute the linkage Can be “euclidean” “l1” “l2” “manhattan” “cosine” or ‘precomputed’ If linkage is “ward” only “euclidean” is accepted

memory None str or object with the joblibMemory interface optional Used to cache the output of the computation of the tree By default no caching is done If a string is given it is the path to the caching directory

connectivity arraylike or callable optional Connectivity matrix Defines for each feature the neighboring features following a given structure of the data This can be a connectivity matrix itself or a callable that transforms the data into a connectivity matrix such as derived from kneighborsgraph Default is None ie the hierarchical clustering algorithm is unstructured

computealltree bool or ‘auto’ optional default “auto” Stop early the construction of the tree at nclusters This is useful to decrease computation time if the number of clusters is not small compared to the number of features This option is useful only when specifying a connectivity matrix Note also that when varying the number of clusters and using caching it may be advantageous to compute the full tree It must be True if distancethreshold is notNone

linkage “ward” “complete” “average” “single” optional default“ward” Which linkage criterion to use The linkage criterion determines which distance to use between sets of features The algorithm will merge the pairs of cluster that minimize this criterion

- ward minimizes the variance of the clusters being merged
- average uses the average of the distances of each feature of the two sets
- complete or maximum linkage uses the maximum distances between all features of the two sets
- single uses the minimum of the distances between all observations of the two sets

poolingfunc callable default npmean This combines the values of agglomerated features into a single value and should accept an array of shape M N and the keyword argument axis1 and reduce it to an array of size M

distancethreshold float optional defaultNone The linkage distance threshold above which clusters will not be merged If not None nclusters must beNone and computealltree must beTrue

New in version 021

Attributes

nclusters int The number of clusters found by the algorithm If

distancethresholdNone it will be equal to the given nclusters

labels arraylike nfeatures cluster labels for each feature

nleaves int Number of leaves in the hierarchical tree

nconnectedcomponents int The estimated number of connected components in the graph

children arraylike shape nnodes1 2 The children of each nonleaf node Values less than nfeatures correspond to leaves of the tree which are the original samples A node 63sklearncluster Clustering 1487

scikitlearn user guide Release 0213

igreater than or equal to nfeatures is a nonleaf node and has children childreni
nfeatures. Alternatively at the ith iteration childreni0 and childreni1 are
merged to form node nfeatures i

Examples

```
import numpy as np
from sklearn import datasets cluster
digits datasetsloaddigits
images digitsimages
X npreshapeimages lenimages 1
agglo clusterFeatureAgglomerationnclusters32
agglofitX
FeatureAgglomerationaffinityeuclidean computefulltreeauto
connectivityNone distancethresholdNone linkageward
memoryNone nclusters32
poolingfunc
Xreduced agglotransformX
Xreducedshape
1797 32
```

Methods

```
fitself X y Fit the hierarchical clustering on the data
fittransform self X y Fit to data then transform it
getparams self deep Get parameters for this estimator
inversetransform self Xred Inverse the transformation
poolingfunc a axis dtype out keepdims Compute the arithmetic mean along the specified axis
setparams self params Set the parameters of this estimator
transform self X Transform a new matrix using the built clustering
init selfnclusters2 affinity'euclidean' memoryNone connectivityNone com
putefulltree'auto' linkage'ward' poolingfuncfunction mean at 0x7f3c23df3400
distancethresholdNone
fitselfXyNone params
```

Fit the hierarchical clustering on the data

Parameters

Xarraylike shape nsamples nfeatures The data

ylgnored

Returns

self

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

scikitlearn user guide Release 0213

numpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfXred

Inverse the transformation Return a vector of size nbfeatures with the values of Xred assigned to each

group of features

Parameters

Xred arraylike shapensamples nclusters or nclusters The values to be assigned

to each cluster of samples

Returns

Xarray shapensamples nfeatures or nfeatures A vector of size nsamples with

the values of Xred assigned to each of the cluster of samples

poolingfunc aaxisNone dtypeNone outNone keepdimsno value

Compute the arithmetic mean along the specified axis

Returns the average of the array elements The average is taken over the flattened array by default other

wise over the specified axis float64 intermediate and return values are used for integer inputs

Parameters

aarraylike Array containing numbers whose mean is desired If ais not an array a

conversion is attempted

axis None or int or tuple of ints optional Axis or axes along which the means are com

puted The default is to compute the mean of the flattened array

New in version 170

If this is a tuple of ints a mean is performed over multiple axes instead of a single axis or

all the axes as before

dtype datatype optional Type to use in computing the mean For integer inputs the

default isfloat64 for floating point inputs it is the same as the input dtype

out ndarray optional Alternate output array in which to place the result The default is

None if provided it must have the same shape as the expected output but the type will

be cast if necessary See docufuncs for details

keepdims bool optional If this is set to True the axes which are reduced are left in the

result as dimensions with size one With this option the result will broadcast correctly

against the input array

If the default value is passed then keepdims will not be passed through to the mean

method of subclasses of ndarray however any nondefault value will be If the sub

class' method does not implement keepdims any exceptions will be raised

63sklearncluster Clustering 1489

scikitlearn user guide Release 0213

Returns

array see dtype parameter above If out=None returns a new array containing the mean values otherwise a reference to the output array is returned

See also

average Weighted average

stdvarnanmean nanstd nanvar

Notes

The arithmetic mean is the sum of the elements along the axis divided by the number of elements

Note that for floatingpoint input the mean is computed using the same precision the input has Depending on the input data this can cause the results to be inaccurate especially for float32 see example below

Specifying a higherprecision accumulator using the dtype keyword can alleviate this issue

By default float16 results are computed using float32 intermediates for extra precision

Examples

```
a = np.array(1 2 3 4)
```

```
np.mean(a)
```

```
2.5
```

```
np.mean(a, axis=0)
```

```
array(2.5)
```

```
np.mean(a, axis=1)
```

```
array(15. 35)
```

In single precision mean can be inaccurate

```
a = np.zeros(2, dtype=np.float32)
```

```
a[0] = 10
```

```
a[1] = 0.1
```

```
np.mean(a)
```

```
0.549999924
```

Computing the mean in float64 is more accurate

```
np.mean(a, dtype=np.float64)
```

```
0.550000000074505806
```

```
set_params(self, params)
```

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform self.X

Transform a new matrix using the built clustering

1490 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures or nfeatures A M by N array of M obser-
vations in N dimensions or a length M array of M onedimensional observations

Returns

Yarray shape nsamples nclusters or nclusters The pooled values for each feature
cluster

Examples using sklearnclusterFeatureAgglomeration

•Feature agglomeration

•Feature agglomeration vs univariate selection

sklearncluster KMeans

classsklearncluster KMeansnclusters8 init'kmeans' ninit10 maxiter300

tol0.00001 precomputeddistances'auto' verbose0 ran-
domstateNone copyxTrue njobsNone algorithm'auto'

KMeans clustering

Read more in the User Guide

Parameters

nclusters int optional default 8 The number of clusters to form as well as the number of
centroids to generate

init 'kmeans' 'random' or an ndarray Method for initialization defaults to 'k-
means'

'kmeans' selects initial cluster centers for kmean clustering in a smart way to speed
up convergence See section Notes in kinit for more details

'random' choose k observations rows at random from data for the initial centroids

If an ndarray is passed it should be of shape nclusters nfeatures and gives the initial
centers

ninit int default 10 Number of time the kmeans algorithm will be run with different cen-
troid seeds The final results will be the best output of ninit consecutive runs in terms of

inertia
maxiter int default 300 Maximum number of iterations of the kmeans algorithm for a
single run

tolfloat default 1e4 Relative tolerance with regards to inertia to declare convergence

precomputeddistances 'auto' True False Precompute distances faster but takes more
memory

'auto' do not precompute distances if nsamples nclusters 12 million This corre-
sponds to about 100MB overhead per job using double precision

True always precompute distances

False never precompute distances

verbose int default 0 Verbosity mode

63sklearncluster Clustering 1491

scikitlearn user guide Release 0213

randomstate int RandomState instance or None default Determines random number generation for centroid initialization Use an int to make the randomness deterministic See Glossary

copyx boolean optional When precomputing distances it is more numerically accurate to center the data first If copyx is True default then the original data is not modified ensuring X is Ccontiguous If False the original data is modified and put back before the function returns but small numerical differences may be introduced by subtracting and then adding the data mean in this case it will also not ensure that data is Ccontiguous which may cause a significant slowdown

njobs int or None optional defaultNone The number of jobs to use for the computation This works by computing each of the ninit runs in parallel

None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

algorithm “auto” “full” or “elkan” default“auto” Kmeans algorithm to use The classical EMstyle algorithm is “full” The “elkan” variation is more efficient by using the triangle inequality but currently doesn’t support sparse data “auto” chooses “elkan” for dense data and “full” for sparse data

Attributes

clustercenters array nclusters nfeatures Coordinates of cluster centers If the algorithm stops before fully converging see tol andmaxiter these will not be consistent withlabels

labels Labels of each point

inertia float Sum of squared distances of samples to their closest cluster center

niter int Number of iterations run

See also

MiniBatchKMeans Alternative online implementation that does incremental updates of the centers positions using minibatches For large scale learning say nsamples 10k MiniBatchKMeans is probably much faster than the default batch implementation

Notes

The kmeans problem is solved using either Lloyd’s or Elkan’s algorithm

The average complexity is given by $O(knT)$ where n is the number of samples and T is the number of iteration

The worst case complexity is given by $O(nk^2p)$ with n nsamples p nfeatures D Arthur and S

Vassilvitskii ‘How slow is the kmeans method’ SoCG2006

In practice the kmeans algorithm is very fast one of the fastest clustering algorithms available but it falls in local minima That’s why it can be useful to restart it several times

If the algorithm stops before fully converging because of tol ormaxiter labels and clustercenters will not be consistent ie the clustercenters will not be the means of the points in each cluster Also the estimator will reassign labels after the last iteration to make labels consistent with predict on the training set

1492 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

```
from sklearncluster import KMeans
import numpy as np
X = np.array([2, 1, 4, 1, 0,
              10, 2, 10, 4, 10, 0])
kmeans = KMeans(nclusters=2, randomstate=0)
fitX
kmeans.labels_
array([1, 1, 1, 0, 0, 0], dtype=int32)
kmeans.predict([0, 0, 12, 3])
array([1, 0], dtype=int32)
kmeans.cluster_centers_
array([10, 2,
        1, 2])
```

Methods

```
fit(self, X, y, sample_weight) Compute kmeans clustering
fit_predict(self, X, y, sample_weight) Compute cluster centers and predict cluster index for
each sample
fit_transform(self, X, y, sample_weight) Compute clustering and transform X to cluster distance
space
get_params(self, deep=True) Get parameters for this estimator
predict(self, X, sample_weight) Predict the closest cluster each sample in X belongs to
score(self, X, y, sample_weight) Opposite of the value of X on the Kmeans objective
set_params(self, **kwargs) Set the parameters of this estimator
transform(self, X) Transform X to a cluster distance space
init(self, nclusters=8, init='kmeans', ninit=10, maxiter=300, tol=0.00001, precompute_distances='auto', verbose=0, randomstate=None, copy_x=True, n_jobs=None, algorithm='auto')
fit(self, X, y, sample_weight=None) Compute kmeans clustering
```

Parameters

X arraylike or sparse matrix shape (n_samples, n_features) Training instances to cluster
It must be noted that the data will be converted to C ordering which will cause a memory copy if the given data is not C contiguous
y ignored not used present here for API (Application Programming Interface) consistency by convention
sample_weight arraylike shape (n_samples,) optional The weights for each observation in X If None all observations are assigned equal weight default None
fit_predict(self, X, y, sample_weight=None) Compute cluster centers and predict cluster index for each sample
Convenience method equivalent to calling fit(X) followed by predict(X)
Parameters
X arraylike sparse matrix shape (n_samples, n_features) New data to transform
y ignored not used present here for API (Application Programming Interface) consistency by convention

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples optional The weights for each observation in X If None all observations are assigned equal weight default None

Returns

labels array shape nsamples Index of the cluster each sample belongs to

fittransform selfXyNone sampleweightNone

Compute clustering and transform X to clusterdistance space

Equivalent to fitXtransformX but more efficiently implemented

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data to transform

ylgnored not used present here for API (Application Programming Interface) consistency by convention

sampleweight arraylike shape nsamples optional The weights for each observation in X If None all observations are assigned equal weight default None

Returns

Xnew array shape nsamples k X transformed in the new space

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXsampleweightNone

Predict the closest cluster each sample in X belongs to

In the vector quantization literature clustercenters is called the code book and each value re

turned bypredict is the index of the closest code in the code book

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data to predict

sampleweight arraylike shape nsamples optional The weights for each observation

in X If None all observations are assigned equal weight default None

Returns

labels array shape nsamples Index of the cluster each sample belongs to

scoreselfXyNone sampleweightNone

Opposite of the value of X on the Kmeans objective

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data

ylgnored not used present here for API (Application Programming Interface) consistency by convention

sampleweight arraylike shape nsamples optional The weights for each observation

in X If None all observations are assigned equal weight default None

Returns

1494 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

score float Opposite of the value of X on the Kmeans objective

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Transform X to a clusterdistance space

In the new space each dimension is the distance to the cluster centers Note that even if X is sparse the array returned by transform will typically be dense

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data to transform

Returns

Xnew array shape nsamples k X transformed in the new space

Examples using sklearnclusterKMeans

- Demonstration of kmeans assumptions
- Vector Quantization Example
- Kmeans Clustering
- Color Quantization using KMeans
- Empirical evaluation of the impact of kmeans initialization
- Comparison of the KMeans and MiniBatchKMeans clustering algorithms
- A demo of KMeans clustering on the handwritten digits data
- Selecting the number of clusters with silhouette analysis on KMeans clustering
- Clustering text documents using kmeans

sklearncluster MiniBatchKMeans

classsklearncluster MiniBatchKMeans nclusters8 init'kmeans' maxiter100

batchsize100 verbose0 computelabelsTrue

randomstateNone tol00 maxnoimprovement10

initsizeNone ninit3 reassignmentratio001

MiniBatch KMeans clustering

Read more in the User Guide

Parameters

nclusters int optional default 8 The number of clusters to form as well as the number of centroids to generate

63sklearncluster Clustering 1495

scikitlearn user guide Release 0213

init 'kmeans' 'random' or an ndarray default 'kmeans' Method for initialization defaults to 'kmeans'

'kmeans' selects initial cluster centers for kmean clustering in a smart way to speed up convergence See section Notes in kinit for more details

'random' choose k observations rows at random from data for the initial centroids

If an ndarray is passed it should be of shape nclusters nfeatures and gives the initial centers

maxiter int optional Maximum number of iterations over the complete dataset before stopping independently of any early stopping criterion heuristics

batchsize int optional default 100 Size of the mini batches

verbose boolean optional Verbosity mode

computelabels boolean default True Compute label assignment and inertia for the complete dataset once the minibatch optimization has converged in fit

randomstate int RandomState instance or None default Determines random number generation for centroid initialization and random reassignment Use an int to make the randomness deterministic See Glossary

tol float default 0.0 Control early stopping based on the relative center changes as measured by a smoothed variancenormalized of the mean center squared position changes This early stopping heuristics is closer to the one used for the batch variant of the algorithms but induces a slight computational and memory overhead over the inertia heuristic

To disable convergence detection based on normalized center change set tol to 0.0 default

maxnoimprovement int default 10 Control early stopping based on the consecutive number of mini batches that does not yield an improvement on the smoothed inertia

To disable convergence detection based on inertia set maxnoimprovement to None

initsize int optional default 3 batchsize Number of samples to randomly sample for speeding up the initialization sometimes at the expense of accuracy the only algorithm is initialized by running a batch KMeans on a random subset of the data This needs to be larger than nclusters

ninit int default 3 Number of random initializations that are tried In contrast to KMeans the algorithm is only run once using the best of the ninit initializations as measured by inertia

reassignmentratio float default 0.01 Control the fraction of the maximum number of counts for a center to be reassigned A higher value means that low count centers are more easily reassigned which means that the model will take longer to converge but should converge in a better clustering

Attributes

clustercenters array nclusters nfeatures Coordinates of cluster centers

labels Labels of each point if computelabels is set to True

inertia float The value of the inertia criterion associated with the chosen partition if computelabels is set to True The inertia is defined as the sum of square distances of samples to their nearest neighbor

See also

1496 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

KMeans The classic implementation of the clustering method based on the Lloyd’s algorithm It consumes the whole set of input data at each iteration

Notes

See <https://www.weecstufstedudsculleypapersfastkmeanspdf>

Examples

```
from sklearn.cluster import MiniBatchKMeans
import numpy as np
X = np.array([2, 1, 4, 1, 0,
               4, 2, 4, 0, 4, 4,
               4, 5, 0, 1, 2, 2,
               3, 2, 5, 5, 1, 1])
```

manually fit on batches

```
kmeans = MiniBatchKMeans(n_clusters=2,
                           random_state=0,
                           batch_size=6)
kmeans.fit(X)
kmeans.predict(X)
kmeans.cluster_centers_
array([1, 3, 4])
```

fit on the whole data

```
kmeans = MiniBatchKMeans(n_clusters=2,
                           random_state=0,
                           batch_size=6,
                           max_iter=10)
kmeans.fit(X)
kmeans.predict(X)
array([0, 4, 4])
```

Methods

- fit(self, X, y=None, sample_weight=None) Compute the centroids on X by chunking it into mini batches
- fit_predict(self, X, y=None, sample_weight=None) Compute cluster centers and predict cluster index for each sample
- fit_transform(self, X, y=None, sample_weight=None) Compute clustering and transform X to clusterdistance space
- get_params(self, deep=True) Get parameters for this estimator
- partial_fit(self, X, y=None, sample_weight=None) Update k means estimate on a single minibatch X
- predict(self, X, y=None, sample_weight=None) Predict the closest cluster each sample in X belongs to
- score(self, X, y=None, sample_weight=None) Opposite of the value of X on the Kmeans objective
- set_params(self, **params) Set the parameters of this estimator
- transform(self, X) Transform X to a clusterdistance space

scikitlearn user guide Release 0213

init selfnclusters8 init'kmeans' maxiter100 batchsize100 verbose0
computelabelsTrue randomstateNone tol00 maxnoimprovement10
initsizeNone ninit3 reassignmentratio001

fitselfXyNone sampleweightNone

Compute the centroids on X by chunking it into minibatches

Parameters

Xarraylike or sparse matrix shapensamples nfeatures Training instances to cluster

It must be noted that the data will be converted to C ordering which will cause a memory

copy if the given data is not Ccontiguous

ylgnored not used present here for API (Application Programming Interface) consistency by convention

sampleweight arraylike shape nsamples optional The weights for each observation

in X If None all observations are assigned equal weight default None

fitpredict selfXyNone sampleweightNone

Compute cluster centers and predict cluster index for each sample

Convenience method equivalent to calling fitX followed by predictX

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data to transform

ylgnored not used present here for API (Application Programming Interface) consistency by convention

sampleweight arraylike shape nsamples optional The weights for each observation

in X If None all observations are assigned equal weight default None

Returns

labels array shape nsamples Index of the cluster each sample belongs to

fittransform selfXyNone sampleweightNone

Compute clustering and transform X to clusterdistance space

Equivalent to fitXtransformX but more efficiently implemented

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data to transform

ylgnored not used present here for API (Application Programming Interface) consistency by convention

sampleweight arraylike shape nsamples optional The weights for each observation

in X If None all observations are assigned equal weight default None

Returns

Xnew array shape nsamples k X transformed in the new space

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyNone sampleweightNone

Update k means estimate on a single minibatch X

1498 Chapter 6 API (Application Programming Interface) Reference

Parameters

Xarraylike shape nsamples nfeatures Coordinates of the data points to cluster It must be noted that X will be copied if it is not Ccontiguous
yIgnored not used present here for API (Application Programming Interface) consistency by convention
sampleweight arraylike shape nsamples optional The weights for each observation in X If None all observations are assigned equal weight default None
predictselfXsampleweightNone

Predict the closest cluster each sample in X belongs to
In the vector quantization literature clustercenters is called the code book and each value returned bypredict is the index of the closest code in the code book

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data to predict
sampleweight arraylike shape nsamples optional The weights for each observation in X If None all observations are assigned equal weight default None
Returns

labels array shape nsamples Index of the cluster each sample belongs to
scoreselfXyNone sampleweightNone

Opposite of the value of X on the Kmeans objective

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data
yIgnored not used present here for API (Application Programming Interface) consistency by convention
sampleweight arraylike shape nsamples optional The weights for each observation in X If None all observations are assigned equal weight default None
Returns

score float Opposite of the value of X on the Kmeans objective
setparams selfparams

Set the parameters of this estimator
The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object
Returns

self
transform selfX

Transform X to a clusterdistance space
In the new space each dimension is the distance to the cluster centers Note that even if X is sparse the array returned by transform will typically be dense

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data to transform
Returns

Xnew array shape nsamples k X transformed in the new space
63sklearncluster Clustering 1499

scikitlearn user guide Release 0213

Examples using sklearnclusterMiniBatchKMeans

- Biclustering documents with the Spectral Coclustering algorithm
- Online learning of a dictionary of parts of faces
- Compare BIRCH and MiniBatchKMeans
- Empirical evaluation of the impact of kmeans initialization
- Comparison of the KMeans and MiniBatchKMeans clustering algorithms
- Comparing different clustering algorithms on toy datasets
- Faces dataset decompositions
- Clustering text documents using kmeans

sklearncluster MeanShift

classssklearncluster MeanShift bandwidthNone seedsNone binseedingFalse

minbinfreq1 clusterallTrue njobsNone

Mean shift clustering using a flat kernel

Mean shift clustering aims to discover “blobs” in a smooth density of samples It is a centroidbased algorithm which works by updating candidates for centroids to be the mean of the points within a given region These candidates are then filtered in a postprocessing stage to eliminate nearduplicates to form the final set of centroids

Seeding is performed using a binning technique for scalability

Read more in the User Guide

Parameters

bandwidth float optional Bandwidth used in the RBF (Radial Basis Function) kernel

If not given the bandwidth is estimated using sklearnclusterestimatebandwidth see the documentation for that function for hints on scalability see also the Notes below

seeds array shapensamples nfeatures optional Seeds used to initialize kernels If not set the seeds are calculated by clusteringgetbinseeds with bandwidth as the grid size and default values for other parameters

binseeding boolean optional If true initial kernel locations are not locations of all points but rather the location of the discretized version of points where points are binned onto a grid whose coarseness corresponds to the bandwidth Setting this option to True will speed up the algorithm because fewer seeds will be initialized default value False Ignored if seeds argument is not None

minbinfreq int optional To speed up the algorithm accept only those bins with at least minbinfreq points as seeds If not defined set to 1

clusterall boolean default True If true then all points are clustered even those orphans that are not within any kernel Orphans are assigned to the nearest kernel If false then orphans are given cluster label 1

njobs int or None optional defaultNone The number of jobs to use for the computation

This works by computing each of the ninit runs in parallel

None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

1500 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Attributes

clustercenters array nclusters nfeatures Coordinates of cluster centers

labels Labels of each point

Notes

Scalability

Because this implementation uses a flat kernel and a Ball Tree to look up members of each kernel the complexity will tend towards $OTn \log n$ in lower dimensions with n the number of samples and T the number of points

In higher dimensions the complexity will tend towards OTn^2

Scalability can be boosted by using fewer seeds for example by using a higher value of minbinfreq in the getbinseeds function

Note that the estimatebandwidth function is much less scalable than the mean shift algorithm and will be the bottleneck if it is used

References

Dorin Comaniciu and Peter Meer “Mean Shift A robust approach toward feature space analysis” IEEE Transactions on Pattern Analysis and Machine Intelligence 2002 pp 603619

Examples

```
from sklearncluster import MeanShift
```

```
import numpy as np
```

```
X = np.array([1, 2, 1, 1, 0,
```

```
4, 7, 3, 5, 3, 6,
```

```
clustering = MeanShiftbandwidth2fit(X,
```

```
clusteringlabels,
```

```
array([1, 1, 0, 0, 0,
```

```
clusteringpredict=[0, 0, 5, 5,
```

```
array([0,
```

```
clustering,
```

```
MeanShiftbandwidth2=binseeding=False clusterall=True minbinfreq=1,
```

```
njobs=None seeds=None)
```

Methods

fitself X y Perform clustering

fitpredict self X y Performs clustering on X and returns cluster labels

getparams self deep Get parameters for this estimator

predict self X Predict the closest cluster each sample in X belongs to

setparams self params Set the parameters of this estimator

init selfbandwidth=None seeds=None binseeding=False minbinfreq=1 clusterall=True

njobs=None

fitselfXy=None

63sklearncluster Clustering 1501

scikitlearn user guide Release 0213

Perform clustering

Parameters

Xarraylike shapensamples nfeatures Samples to cluster

ylgnored

fitpredict selfXyNone

Performs clustering on X and returns cluster labels

Parameters

Xndarray shape nsamples nfeatures Input data

ylgnored not used present for API (Application Programming Interface) consistency by convention

Returns

labels ndarray shape nsamples cluster labels

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict the closest cluster each sample in X belongs to

Parameters

Xarraylike sparse matrix shapensamples nfeatures New data to predict

Returns

labels array shape nsamples Index of the cluster each sample belongs to

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnclusterMeanShift

•A demo of the meanshift clustering algorithm

•Comparing different clustering algorithms on toy datasets

1502 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearncluster SpectralClustering

class sklearncluster SpectralClustering nclusters8 eigensolverNone ran

domstateNone ninit10 gamma10 affin

ity'rbf' nneighbors10 eigentol00 as

signlabels'kmeans' degree3 coef01 ker

nelparamsNone njobsNone

Apply clustering to a projection of the normalized Laplacian

In practice Spectral Clustering is very useful when the structure of the individual clusters is highly nonconvex or more generally when a measure of the center and spread of the cluster is not a suitable description of the complete cluster For instance when clusters are nested circles on the 2D plane

If affinity is the adjacency matrix of a graph this method can be used to find normalized graph cuts

When calling fit an affinity matrix is constructed using either kernel function such the Gaussian aka RBF (Radial Basis Function) kernel of the euclidean distanced dX X

npexpgamma dXX2

or a knearest neighbors connectivity matrix

Alternatively using precomputed a userprovided affinity matrix can be used

Read more in the User Guide

Parameters

nclusters integer optional The dimension of the projection subspace

eigensolver None 'arpack' 'lobpcg' or 'amg' The eigenvalue decomposition strategy to use AMG requires pyamg to be installed It can be faster on very large sparse problems but may also lead to instabilities

randomstate int RandomState instance or None default A pseudo random number generator used for the initialization of the lobpcg eigen vectors decomposition when eigensolveramg and by the KMeans initialization Use an int to make the ran

domness deterministic See Glossary

ninit int optional default 10 Number of time the kmeans algorithm will be run with dif

ferent centroid seeds The final results will be the best output of ninit consecutive runs in

terms of inertia

gamma float default10 Kernel coefficient for rbf poly sigmoid laplacian and chi2 kernels

Ignored for affinitynearestneighbors

affinity string arraylike or callable default 'rbf' If a string this may be one of 'near

estneighbors' 'precomputed' 'rbf' or one of the kernels supported by sklearn

metricspairwise kernels

Only kernels that produce similarity scores nonnegative values that increase with similar

ity should be used This property is not checked by the clustering algorithm

nneighbors integer Number of neighbors to use when constructing the affinity matrix using

the nearest neighbors method Ignored for affinityrbf

eigentol float optional default 00 Stopping criterion for eigendecomposition of the Lapla

cian matrix when eigensolverarpack

assignlabels 'kmeans' 'discretize' default 'kmeans' The strategy to use to assign labels

in the embedding space There are two ways to assign labels after the laplacian embedding

63sklearncluster Clustering 1503

scikitlearn user guide Release 0213

kmeans can be applied and is a popular choice But it can also be sensitive to initialization

Discretization is another approach which is less sensitive to random initialization

degree float default3 Degree of the polynomial kernel Ignored by other kernels

coef0 float default1 Zero coefficient for polynomial and sigmoid kernels Ignored by other kernels

kernelparams dictionary of string to any optional Parameters keyword arguments and values for kernel passed as callable object Ignored by other kernels

njobs int or None optional defaultNone The number of parallel jobs to run None means

1 unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

Attributes

affinitymatrix arraylike shape nsamples nsamples Affinity matrix used for cluster

ing Available only if after calling fit

labels Labels of each point

Notes

If you have an affinity matrix such as a distance matrix for which 0 means identical elements and high values

means very dissimilar elements it can be transformed in a similarity matrix that is well suited for the algorithm

by applying the Gaussian RBF (Radial Basis Function) heat kernel

npexp distmatrix 2 2 delta2

Wheredelta is a free parameter representing the width of the Gaussian kernel

Another alternative is to take a symmetric version of the k nearest neighbors connectivity matrix of the points

If the pyamg package is installed it is used this greatly speeds up computation

References

- Normalized cuts and image segmentation 2000 Jianbo Shi Jitendra Malik <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.160.2324>

- A Tutorial on Spectral Clustering 2007 Ulrike von Luxburg <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.165.9323>

- Multiclass spectral clustering 2003 Stella X Yu Jianbo Shi <https://www1ics.berkeley.edu/stellayu/publication/doc2003/kwayl/ICCV.pdf>

Examples

```
from sklearn.cluster import SpectralClustering
```

```
import numpy as np
```

```
X = np.array([1 2 1 1 0
```

```
4 7 3 5 3 6
```

```
clustering = SpectralClustering(n_clusters=2
```

```
assign_labels=True
```

```
random_state=0)
```

```
clustering.labels
```

1504 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

array1 1 1 0 0 0

clustering

SpectralClusteringaffinityrbf assignlabelsdiscretize coef01

degree3 eigensolverNone eigentol00 gamma10

kernelparamsNone nclusters2 ninit10 njobsNone

nneighbors10 randomstate0

Methods

fitself X y Creates an affinity matrix for X using the selected affinity then applies spectral clustering to this affinity matrix

fitpredict self X y Performs clustering on X and returns cluster labels

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

init selfnclusters8 eigensolverNone randomstateNone ninit10 gamma10 affinity'rbf' nneighbors10 eigentol00 assignlabels'kmeans' degree3 coef01

kernelparamsNone njobsNone

fitselfXyNone

Creates an affinity matrix for X using the selected affinity then applies spectral clustering to this affinity matrix

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures OR if affinity'precomputed' a precomputed affinity matrix of shape nsamples nsamples

ylgnored

fitpredict selfXyNone

Performs clustering on X and returns cluster labels

Parameters

Xndarray shape nsamples nfeatures Input data

ylgnored not used present for API (Application Programming Interface) consistency by convention

Returns

labels ndarray shape nsamples cluster labels

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

63sklearncluster Clustering 1505

scikitlearn user guide Release 0213

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnclusterSpectralClustering

•Comparing different clustering algorithms on toy datasets

632 Functions

clusteraffinitypropagation S Perform Affinity Propagation Clustering of data

clusterclusteropticsdbscan reachability

Performs DBSCAN extraction for an arbitrary epsilon

clusterclusteropticsxi reachability Automatically extract clusters according to the Xisteeep

method

clustercomputeopticsgraph X

minsamples Computes the OPTICS reachability graph

clusterdbscan X eps minsamples Perform DBSCAN clustering from vector array or distance

matrix

clusterestimatebandwidth X quantile Estimate the bandwidth to use with the meanshift algo

rithm

clusterkmeans X nclusters Kmeans clustering algorithm

clustermeanshift X bandwidth seeds Perform mean shift clustering of data using a flat kernel

clusterspectralclustering affinity Apply clustering to a projection of the normalized Lapla

cian

clusterwardtree X connectivity Ward clustering based on a Feature matrix

sklearncluster affinitypropagation

sklearncluster affinitypropagation S preferenceNone convergenceiter15

maxiter200 damping05 copyTrue ver

boseFalse returnniterFalse

Perform Affinity Propagation Clustering of data

Read more in the User Guide

Parameters

Sarraylike shape nsamples nsamples Matrix of similarities between points

preference arraylike shape nsamples or float optional Preferences for each point points

with larger values of preferences are more likely to be chosen as exemplars The number of

exemplars ie of clusters is influenced by the input preferences value If the preferences

are not passed as arguments they will be set to the median of the input similarities resulting

in a moderate number of clusters For a smaller amount of clusters this can be set to the

minimum value of the similarities

convergenceiter int optional default 15 Number of iterations with no change in the number

of estimated clusters that stops the convergence

maxiter int optional default 200 Maximum number of iterations

1506 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

damping float optional default 0.5 Damping factor between 0.5 and 1

copy boolean optional default True If copy is False the affinity matrix is modified inplace by the algorithm for memory efficiency

verbose boolean optional default False The verbosity level

return_niter bool default False Whether or not to return the number of iterations

Returns

cluster_centers indices array shape n_clusters index of clusters centers

labels array shape n_samples cluster labels for each point

niter int number of iterations run Returned only if return_niter is set to True

Notes

For an example see examples/cluster/plot_affinity_propagation.py

When the algorithm does not converge it returns an empty array as cluster_centers and 1 as label for each training sample

When all training samples have equal similarities and equal preferences the assignment of cluster centers and labels depends on the preference. If the preference is smaller than the similarities a single cluster center and label 0 for every sample will be returned. Otherwise every training sample becomes its own cluster center and is assigned a unique label.

References

Brendan J. Frey and Delbert Dueck “Clustering by Passing Messages Between Data Points” Science Feb 2007

Examples using sklearn.cluster.affinity_propagation

• Visualizing the stock market structure

sklearn.cluster.cluster_optics_dbscan

sklearn.cluster.cluster_optics_dbscan.reachability_core_distances ordering eps

Performs DBSCAN extraction for an arbitrary epsilon

Extracting the clusters runs in linear time. Note that this results in labels which are close to a DBSCAN with similar settings and epsilon only if epsilon is close to max_eps.

Parameters

reachability array shape n_samples Reachability distances calculated by OPTICS

reachability

core_distances array shape n_samples Distances at which points become core

core_distances

ordering array shape n_samples OPTICS ordered point indices ordering

eps float DBSCAN epsilon parameter Must be set to max_eps. Results will be close to

DBSCAN algorithm if epsilon and max_eps are close to one another

63 sklearn.cluster Clustering 1507

scikitlearn user guide Release 0213

Returns

labels array shape nsamples The estimated labels

Examples using sklearnclusterclusteropticsdbscan

•Demo of OPTICS clustering algorithm

sklearncluster clusteropticsxi

sklearncluster clusteropticsxi reachability predecessor ordering minsamples

minclustersizeNone xi005 predeces

sorrectionTrue

Automatically extract clusters according to the Xisteeep method

Parameters

reachability array shape nsamples Reachability distances calculated by OPTICS

reachability

predecessor array shape nsamples Predecessors calculated by OPTICS

ordering array shape nsamples OPTICS ordered point indices ordering

minsamples int 1 or float between 0 and 1 The same as the minsamples given to OPTICS

Up and down steep regions can't have more then minsamples consecutive nonsteep

points Expressed as an absolute number or a fraction of the number of samples rounded to be at least 2

minclustersize int 1 or float between 0 and 1 defaultNone Minimum number of sam

ples in an OPTICS cluster expressed as an absolute number or a fraction of the number of

samples rounded to be at least 2 If None the value of minsamples is used instead

xifloat between 0 and 1 optional default005 Determines the minimum steepness on the

reachability plot that constitutes a cluster boundary For example an upwards point in the

reachability plot is defined by the ratio from one point to its successor being at most 1xi

predecessorcorrection bool optional defaultTrue Correct clusters based on the calcu

lated predecessors

Returns

labels array shape nsamples The labels assigned to samples Points which are not included in any cluster are labeled as 1

clusters array shape nclusters 2 The list of clusters in the form of start end

in each row with all indices inclusive The clusters are ordered according to end

start ascending so that larger clusters encompassing smaller clusters come af

ter such nested smaller clusters Since labels does not reflect the hierarchy usually

lenclusters npuniquelabels

sklearncluster computeopticsgraph

sklearncluster computeopticsgraph Xminsamples maxeps metric pmetricparams al

gorithm leafsize njobs

Computes the OPTICS reachability graph

Read more in the User Guide

1508 Chapter 6 API (Application Programming Interface) Reference

Parameters

Xarray shape nsamples nfeatures or nsamples nsamples if metric='precomputed'

A feature array or array of distances between samples if metric='precomputed'

minsamples int 1 or float between 0 and 1 The number of samples in a neighborhood for a point to be considered as a core point Expressed as an absolute number or a fraction of the number of samples rounded to be at least 2

maxeps float optional default npinf The maximum distance between two samples for one to be considered as in the neighborhood of the other Default value of npinf will identify clusters across all scales reducing maxeps will result in shorter run times

metric string or callable optional default 'minkowski' Metric to use for distance computation Any metric from scikitlearn or scipyspatialdistance can be used

If metric is a callable function it is called on each pair of instances rows and the resulting value recorded The callable should take two arrays as input and return one value indicating the distance between them This works for Scipy's metrics but is less efficient than passing the metric name as a string If metric is "precomputed" X is assumed to be a distance matrix and must be square

Valid values for metric are

- from scikitlearn 'cityblock' 'cosine' 'euclidean' 'l1' 'l2' 'manhattan'
- from scipyspatialdistance 'braycurtis' 'canberra' 'chebyshev' 'correlation' 'dice' 'hamming' 'jaccard' 'kulsinski' 'mahalanobis' 'minkowski' 'rogerstanimoto' 'rusellrao' 'seuclidean' 'sokalmichener' 'sokalsneath' 'sqeuclidean' 'yule'

See the documentation for scipyspatialdistance for details on these metrics

p integer optional default 2 Parameter for the Minkowski metric from sklearn metricspairwisel1distances When p 1 this is equivalent to using manhattan distance l1 and euclidean distance l2 for p 2 For arbitrary p minkowski distance lp is used

metricparams dict optional default None Additional keyword arguments for the metric function

algorithm 'auto' 'balltree' 'kdtree' 'brute' optional Algorithm used to compute the nearest neighbors

- 'balltree' will use BallTree
- 'kdtree' will use KDTree
- 'brute' will use a brute force search
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed

tofit method default

Note fitting on sparse input will override the setting of this parameter using brute force

leafsize int optional default 30 Leaf size passed to BallTree or KDTree This can affect the speed of the construction and query as well as the memory required to store the tree The optimal value depends on the nature of the problem

njobs int or None optional default None The number of parallel jobs to run for neighbors

search None means 1 unless in a joblibparallelbackend context 1 means using all processors See Glossary for more details

Returns

63sklearncluster Clustering 1509

scikitlearn user guide Release 0213

ordering array shape nsamples The cluster ordered list of sample indices
coredistances array shape nsamples Distance at which each sample becomes a core
point indexed by object order Points which will never be core have a distance of inf Use
clustcoredistancesclustordering to access in cluster order
reachability array shape nsamples Reachability distances per sample indexed by object
order Useclustreachabilityclustordering to access in cluster order
predecessor array shape nsamples Point that a sample was reached from indexed by
object order Seed points have a predecessor of 1

References

1

sklearncluster dbscan
sklearncluster dbscanXeps05 minsamples5 metric'minkowski' metricparamsNone al
gorithm'auto' leafsize30 p2sampleweightNone njobsNone
Perform DBSCAN clustering from vector array or distance matrix
Read more in the User Guide

Parameters

Xarray or sparse CSR (Compressed Sparse Row) matrix of shape nsamples nfeatures or array of shape
nsamples nsamples A feature array or array of distances between samples if
metricprecomputed
eps float optional The maximum distance between two samples for one to be considered as
in the neighborhood of the other This is not a maximum bound on the distances of points
within a cluster This is the most important DBSCAN parameter to choose appropriately for
your data set and distance function
minsamples int optional The number of samples or total weight in a neighborhood for a
point to be considered as a core point This includes the point itself
metric string or callable The metric to use when calculating distance between instances in
a feature array If metric is a string or callable it must be one of the options allowed
bysklearnmetricspairwiselocaldistances for its metric parameter If metric
is "precomputed" X is assumed to be a distance matrix and must be square X may be
a sparse matrix in which case only "nonzero" elements may be considered neighbors for
DBSCAN
metricparams dict optional Additional keyword arguments for the metric function

New in version 0.19

algorithm 'auto' 'balltree' 'kdtree' 'brute' optional The algorithm to be used by the
NearestNeighbors module to compute pointwise distances and find nearest neighbors See
NearestNeighbors module documentation for details
leafsize int optional default 30 Leaf size passed to BallTree or cKDTree This can affect
the speed of the construction and query as well as the memory required to store the tree
The optimal value depends on the nature of the problem
pfloat optional The power of the Minkowski metric to be used to calculate distance between
points

scikitlearn user guide Release 0213

sampleweight array shape nsamples optional Weight of each sample such that a sample with a weight of at least minsamples is by itself a core sample a sample with negative weight may inhibit its epsneighbor from being core Note that weights are absolute and default to 1

njobs int or None optional defaultNone The number of parallel jobs to run for neighbors

searchNone means 1 unless in a joblibparallelbackend context1means

using all processors See Glossary for more details

Returns

coresamples array ncoresamples Indices of core samples

labels array nsamples Cluster labels for each point Noisy samples are given the label 1

See also

DBSCAN An estimator interface for this clustering algorithm

OPTICS A similar estimator interface clustering at multiple values of eps Our implementation is optimized for memory usage

Notes

For an example see examplesclusterplotdbscanpy

This implementation bulkcomputes all neighborhood queries which increases the memory complexity to

On d where d is the average number of neighbors while original DBSCAN had memory complexity On

It may attract a higher memory complexity when querying these nearest neighborhoods depending on the algorithm

One way to avoid the query complexity is to precompute sparse neighborhoods in chunks

usingNearestNeighborsradiusneighborsgraph withmodedistance then using

metricprecomputed here

Another way to reduce memory and computation time is to remove nearduplicate points and use

sampleweight instead

clusteroptics provides a similar clustering with lower memory usage

References

Ester M H P Kriegel J Sander and X Xu “A DensityBased Algorithm for Discovering Clusters in Large Spatial Databases with Noise” In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining Portland OR AAAI Press pp 226231 1996

Schubert E Sander J Ester M Kriegel H P Xu X 2017 DBSCAN revisited revisited why and how you should still use DBSCAN ACM Transactions on Database Systems TODS 423 19

sklearncluster estimatebandwidth

sklearncluster estimatebandwidth Xquantile03 nsamplesNone randomstate0

njobsNone

Estimate the bandwidth to use with the meanshift algorithm

That this function takes time at least quadratic in nsamples For large datasets it’s wise to set that parameter to a small value

63sklearncluster Clustering 1511

scikitlearn user guide Release 0213

Parameters

Xarraylike shapensamples nfeatures Input points
quantile float default 0.3 should be between 0 1 0.5 means that the median of all pairwise distances is used
nsamples int optional The number of samples to use If not given all samples are used
randomstate int RandomState instance or None default The generator used to randomly select the samples from input points for bandwidth estimation Use an int to make the randomness deterministic See Glossary
njobs int or None optional defaultNone The number of parallel jobs to run for neighbors
searchNone means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

Returns

bandwidth float The bandwidth parameter
Examples using sklearnclusterestimatebandwidth
•A demo of the meanshift clustering algorithm
•Comparing different clustering algorithms on toy datasets

sklearncluster kmeans
sklearncluster kmeansXnclusters sampleweightNone init'kmeans' precomputeddistances'auto' ninit10 maxiter300 verboseFalse
tol0.0001 randomstateNone copyxTrue njobsNone algo
rithm'auto' returnniterFalse
Kmeans clustering algorithm
Read more in the User Guide

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The observations to cluster It must be noted that the data will be converted to C ordering which will cause a memory copy if the given data is not Ccontiguous
nclusters int The number of clusters to form as well as the number of centroids to generate
sampleweight arraylike shape nsamples optional The weights for each observation in X If None all observations are assigned equal weight default None
init 'kmeans' 'random' or ndarray or a callable optional Method for initialization default to 'kmeans'
'kmeans' selects initial cluster centers for kmean clustering in a smart way to speed up convergence See section Notes in kinit for more details
'random' choose k observations rows at random from data for the initial centroids
If an ndarray is passed it should be of shape nclusters nfeatures and gives the initial centers
If a callable is passed it should take arguments X k and and a random state and return an initialization

scikitlearn user guide Release 0213

precomputeddistances 'auto' True False Precompute distances faster but takes more memory

'auto' do not precompute distances if nsamples nclusters 12 million This corresponds to about 100MB overhead per job using double precision

True always precompute distances

False never precompute distances

ninit int optional default 10 Number of time the kmeans algorithm will be run with different centroid seeds The final results will be the best output of ninit consecutive runs in terms of inertia

maxiter int optional default 300 Maximum number of iterations of the kmeans algorithm to run

verbose boolean optional Verbosity mode

tolfloat optional The relative increment in the results before declaring convergence

randomstate int RandomState instance or None default Determines random number generation for centroid initialization Use an int to make the randomness deterministic See Glossary

copyx boolean optional When precomputing distances it is more numerically accurate to center the data first If copyx is True default then the original data is not modified ensuring X is Ccontiguous If False the original data is modified and put back before the function returns but small numerical differences may be introduced by subtracting and then adding the data mean in this case it will also not ensure that data is Ccontiguous which may cause a significant slowdown

njobs int or None optional defaultNone The number of jobs to use for the computation This works by computing each of the ninit runs in parallel

None means 1 unless in a joblibparallelbackend context1means using all

processors See Glossary for more details

algorithm "auto" "full" or "elkan" default"auto" Kmeans algorithm to use The classical EMstyle algorithm is "full" The "elkan" variation is more efficient by using the triangle inequality but currently doesn't support sparse data "auto" chooses "elkan" for dense data and "full" for sparse data

returnniter bool optional Whether or not to return the number of iterations

Returns

centroid float ndarray with shape k nfeatures Centroids found at the last iteration of k means

label integer ndarray with shape nsamples labeli is the code or index of the centroid the i'th observation is closest to

inertia float The final value of the inertia criterion sum of squared distances to the closest centroid for all observations in the training set

bestniter int Number of iterations corresponding to the best results Returned only if returnniter is set to True

63sklearncluster Clustering 1513

scikitlearn user guide Release 0213

sklearncluster meanshift

sklearncluster meanshift X bandwidthNone seedsNone binseedingFalse

minbinfreq1 clusterallTrue maxiter300 njobsNone

Perform mean shift clustering of data using a flat kernel

Read more in the User Guide

Parameters

Xarraylike shapensamples nfeatures Input data

bandwidth float optional Kernel bandwidth

If bandwidth is not given it is determined using a heuristic based on the median of all pairwise distances This will take quadratic time in the number of samples The sklearnclusterestimatebandwidth function can be used to do this more efficiently seeds arraylike shapenseeds nfeatures or None Point used as initial kernel locations If None and binseedingFalse each data point is used as a seed If None and binseedingTrue see binseeding binseeding boolean defaultFalse If true initial kernel locations are not locations of all points but rather the location of the discretized version of points where points are binned onto a grid whose coarseness corresponds to the bandwidth Setting this option to True will speed up the algorithm because fewer seeds will be initialized Ignored if seeds argument is not None

minbinfreq int default1 To speed up the algorithm accept only those bins with at least

minbinfreq points as seeds

clusterall boolean default True If true then all points are clustered even those orphans that are not within any kernel Orphans are assigned to the nearest kernel If false then orphans are given cluster label 1

maxiter int default 300 Maximum number of iterations per seed point before the clustering operation terminates for that seed point if has not converged yet

njobs int or None optional defaultNone The number of jobs to use for the computation

This works by computing each of the ninit runs in parallel

None means 1 unless in a joblibparallelbackend context1means using all

processors See Glossary for more details

New in version 017 Parallel Execution using njobs

Returns

clustercenters array shapenclusters nfeatures Coordinates of cluster centers

labels array shapensamples Cluster labels for each point

Notes

For an example see examplesclusterplotmeanshift.py

1514 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearncluster spectralclustering

sklearncluster spectralclustering affinity nclusters8 ncomponentsNone

eigensolverNone randomstateNone ninit10

eigentol00 assignlabels'kmeans'

Apply clustering to a projection of the normalized Laplacian

In practice Spectral Clustering is very useful when the structure of the individual clusters is highly nonconvex or more generally when a measure of the center and spread of the cluster is not a suitable description of the complete cluster For instance when clusters are nested circles on the 2D plane

If affinity is the adjacency matrix of a graph this method can be used to find normalized graph cuts

Read more in the User Guide

Parameters

affinity arraylike or sparse matrix shape nsamples nsamples The affinity matrix describing the relationship of the samples to embed Must be symmetric

Possible examples

- adjacency matrix of a graph
- heat kernel of the pairwise distance matrix of the samples
- symmetric knearest neighbours connectivity matrix of the samples

nclusters integer optional Number of clusters to extract

ncomponents integer optional default is nclusters Number of eigen vectors to use for the spectral embedding

eigensolver None 'arpack' 'lobpcg' or 'amg' The eigenvalue decomposition strategy to use AMG requires pyamg to be installed It can be faster on very large sparse problems but may also lead to instabilities

randomstate int RandomState instance or None default A pseudo random number generator used for the initialization of the lobpcg eigen vectors decomposition when eigensolver 'amg' and by the KMeans initialization Use an int to make the randomness deterministic See Glossary

ninit int optional default 10 Number of time the kmeans algorithm will be run with different centroid seeds The final results will be the best output of ninit consecutive runs in terms of inertia

eigentol float optional default 00 Stopping criterion for eigendecomposition of the Laplacian matrix when using arpack eigensolver

assignlabels 'kmeans' 'discretize' default 'kmeans' The strategy to use to assign labels in the embedding space There are two ways to assign labels after the laplacian embedding kmeans can be applied and is a popular choice But it can also be sensitive to initialization Discretization is another approach which is less sensitive to random initialization See the 'Multiclass spectral clustering' paper referenced below for more details on the discretization approach

Returns

labels array of integers shape nsamples The labels of the clusters

63sklearncluster Clustering 1515

scikitlearn user guide Release 0213

Notes

The graph should contain only one connect component elsewhere the results make little sense
This algorithm solves the normalized cut for k2 it is a normalized spectral clustering

References

- Normalized cuts and image segmentation 2000 Jianbo Shi Jitendra Malik <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.160.2324>
- A Tutorial on Spectral Clustering 2007 Ulrike von Luxburg <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.165.9323>
- Multiclass spectral clustering 2003 Stella X Yu Jianbo Shi <https://www1ics.berkeley.edu/stellayu/publication/doc2003/kwayl/ICCV.pdf>

Examples using sklearn.cluster.spectral_clustering

- Segmenting the picture of greek coins in regions
- Spectral clustering for image segmentation

sklearn.cluster.WardTree

sklearn.cluster.WardTree(X, connectivity=None, n_clusters=None, return_distance=False)

Ward clustering based on a Feature matrix

Recursively merges the pair of clusters that minimally increases within cluster variance

The inertia matrix uses a Heapq-based representation

This is the structured version that takes into account some topological structure between samples

Read more in the User Guide

Parameters

X: array shape (n_samples, n_features) feature matrix representing n_samples samples to be clustered

connectivity: sparse matrix optional connectivity matrix Defines for each sample the neighboring samples following a given structure of the data The matrix is assumed to be symmetric and only the upper triangular half is used Default is None ie the Ward algorithm is unstructured

n_clusters: int optional Stop early the construction of the tree at n_clusters This is useful to decrease computation time if the number of clusters is not small compared to the number of samples In this case the complete tree is not computed thus the ‘children’ output is of limited use and the ‘parents’ output should rather be used This option is valid only when specifying a connectivity matrix

return_distance: bool optional If True return the distance between the clusters

Returns

1516 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

children 2D array shape nnodes1 2 The children of each nonleaf node Values less than nsamples correspond to leaves of the tree which are the original samples A node igreater than or equal to nsamples is a nonleaf node and has children childreni nsamples Alternatively at the ith iteration childreni0 and childreni1 are merged to form node nsamples i
nconnectedcomponents int The number of connected components in the graph
nleaves int The number of leaves in the tree
parents 1D array shape nnodes or None The parent of each node Only returned when a connectivity matrix is specified elsewhere 'None' is returned
distances 1D array shape nnodes1 Only returned if returndistance is set to True for compatibility The distances between the centers of the nodes distancesi corresponds to a weighted euclidean distance between the nodes childreni 1 and childreni 2 If the nodes refer to leaves of the tree then distancesi is their unweighted euclidean distance Distances are updated in the following way from scipyhierarchylinkage

The new entry d_{i+1} is computed as follows

$$d_{i+1} = \sqrt{\frac{d_{i+1,0}^2 + d_{i+1,1}^2}{2} - \frac{d_{i+1,0}^2 + d_{i+1,1}^2}{2}}$$

where d_{i+1} is the newly joined cluster consisting of clusters $d_{i+1,0}$ and $d_{i+1,1}$ is an unused cluster in the forest $d_{i+1,0}$ and $d_{i+1,1}$ is the cardinality of its argument This is also known as the incremental algorithm

64sklearnclusterbicluster Biclustering

Spectral biclustering algorithms

Authors Kemal Eren License BSD (Binary Search Decision) 3 clause

User guide See the Biclustering section for further details

641 Classes

SpectralBiclustering nclusters method Spectral biclustering Kluger 2003

SpectralCoclustering nclusters Spectral CoClustering algorithm Dhillon 2001

sklearnclusterbicluster SpectralBiclustering

classsklearnclusterbicluster SpectralBiclustering nclusters3

method'bistochastic'

ncomponents6 nbest3

svdmethod'randomized'

nsvdvecsNone

minibatchFalse init'k

means' ninit10

njobsNone ran

domstateNone

Spectral biclustering Kluger 2003

64sklearnclusterbicluster Biclustering 1517

scikitlearn user guide Release 0213

Partitions rows and columns under the assumption that the data has an underlying checkerboard structure For instance if there are two row partitions and three column partitions each row will belong to three biclusters and each column will belong to two biclusters The outer product of the corresponding row and column label vectors gives this checkerboard structure

Read more in the User Guide

Parameters

nclusters integer or tuple nrowclusters ncolumnclusters The number of row and column clusters in the checkerboard structure

method string optional default 'bistochastic' Method of normalizing and converting singular vectors into biclusters May be one of 'scale' 'bistochastic' or 'log' The authors recommend using 'log' If the data is sparse however log normalization will not work which is why the default is 'bistochastic' CAUTION if methodlog the data must not be sparse

ncomponents integer optional default 6 Number of singular vectors to check

nbest integer optional default 3 Number of best singular vectors to which to project the data for clustering

svdmethod string optional default 'randomized' Selects the algorithm for finding singular vectors May be 'randomized' or 'arpack' If 'randomized' uses sklearnutils extmathrandomizedsvd which may be faster for large matrices If 'arpack' uses scipysparselinalgsvds which is more accurate but possibly slower in some cases

nsvdvecs int optional default None Number of vectors to use in calculating the SVD Corresponds to ncv whensvdmethodarpack andnoversamples when svdmethod is 'randomized'

minibatch bool optional default False Whether to use minibatch kmeans which is faster but may get different results

init 'kmeans' 'random' or an ndarray Method for initialization of kmeans algorithm defaults to 'kmeans'

ninit int optional default 10 Number of random initializations that are tried with the k means algorithm

If minibatch kmeans is used the best initialization is chosen and the algorithm runs once Otherwise the algorithm is run for each initialization and the best solution chosen

njobs int or None optional defaultNone The number of jobs to use for the computation This works by breaking down the pairwise matrix into njobs even slices and computing them in parallel

None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

randomstate int RandomState instance or None default Used for randomizing the singular value decomposition and the kmeans initialization Use an int to make the randomness deterministic See Glossary

Attributes

rows arraylike shape nrowclusters nrow Results of the clustering rowsi r is

True if cluster icontains row r Available only after calling fit

columns arraylike shape ncolumnclusters ncolumn Results of the clustering like rows

1518 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

rowlabels arraylike shape nrows Row partition labels

columnlabels arraylike shape ncols Column partition labels

References

- Kluger Yuval et al 2003 Spectral biclustering of microarray data coclustering genes and conditions

Examples

```
from sklearncluster import SpectralBiclustering
```

```
import numpy as np
```

```
X nparray1 1 2 1 1 0
```

```
4 7 3 5 3 6
```

```
clustering SpectralBiclusteringnclusters2 randomstate0fitX
```

```
clusteringrowlabels
```

```
array1 1 1 0 0 0 dtypeint32
```

```
clusteringcolumnlabels
```

```
array0 1 dtypeint32
```

```
clustering
```

```
SpectralBiclusteringinitkmeans methodbistochastic
```

```
minibatchFalse nbest3 nclusters2 ncomponents6
```

```
ninit10 njobsNone nsvdvecsNone randomstate0
```

```
svdmethodrandomized
```

Methods

```
fitself X y Creates a biclustering for X
```

```
getindices self i Row and column indices of the i'th bicluster
```

```
getparams self deep Get parameters for this estimator
```

```
getshape self i Shape of the i'th bicluster
```

```
getsubmatrix self i data Returns the submatrix corresponding to bicluster i
```

```
setparams self params Set the parameters of this estimator
```

```
init self nclusters3 method'bistochastic' ncomponents6 nbest3
```

```
svdmethod'randomized' nsvdvecsNone minibatchFalse init'kmeans'
```

```
ninit10 njobsNone randomstateNone
```

```
biclusters
```

Convenient way to get row and column indicators together

Returns the rows andcolumns members

```
fitselfXyNone
```

Creates a biclustering for X

Parameters

Xarraylike shape nsamples nfeatures

ylgnored

getindices selfi

Row and column indices of the i'th bicluster

64sklearnclusterbicluster Biclustering 1519

scikitlearn user guide Release 0213

Only works if rows andcolumns attributes exist

Parameters

iint The index of the cluster

Returns

rowind nparray dtypepintp Indices of rows in the dataset that belong to the bicluster

colind nparray dtypepintp Indices of columns in the dataset that belong to the biclus
ter

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getshape selfi

Shape of the i'th bicluster

Parameters

iint The index of the cluster

Returns

shape int int Number of rows and columns resp in the bicluster

getsubmatrix selfidata

Returns the submatrix corresponding to bicluster i

Parameters

iint The index of the cluster

data array The data

Returns

submatrix array The submatrix corresponding to bicluster i

Notes

Works with sparse matrices Only works if rows andcolumns attributes exist

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it's possible to update each component
of a nested object

Returns

self

1520 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearnclusterbicluster SpectralCoclustering

classsklearnclusterbicluster SpectralCoclustering nclusters3

svdmethod'randomized'

nsvdvecsNone

minibatchFalse init'k

means' ninit10

njobsNone ran

domstateNone

Spectral CoClustering algorithm Dhillon 2001

Clusters rows and columns of an array Xto solve the relaxed normalized cut of the bipartite graph created from

Xas follows the edge between row vertex iand column vertex jhas weight X_{ij}

The resulting bicluster structure is blockdiagonal since each row and each column belongs to exactly one

bicluster

Supports sparse matrices as long as they are nonnegative

Read more in the User Guide

Parameters

nclusters integer optional default 3 The number of biclusters to find

svdmethod string optional default 'randomized' Selects the algorithm for finding singu

lar vectors May be 'randomized' or 'arpack' If 'randomized' use sklearnutils

extmathrandomizedsvd which may be faster for large matrices If 'arpack' use

scipysparselinalgsvds which is more accurate but possibly slower in some

cases

nsvdvecs int optional default None Number of vectors to use in calculating the

SVD Corresponds to ncv whensvdmethodarpack andnoversamples when

svdmethod is 'randomized'

minibatch bool optional default False Whether to use minibatch kmeans which is faster

but may get different results

init 'kmeans' 'random' or an ndarray Method for initialization of kmeans algorithm

defaults to 'kmeans'

ninit int optional default 10 Number of random initializations that are tried with the k

means algorithm

If minibatch kmeans is used the best initialization is chosen and the algorithm runs once

Otherwise the algorithm is run for each initialization and the best solution chosen

njobs int or None optional defaultNone The number of jobs to use for the computation

This works by breaking down the pairwise matrix into njobs even slices and computing

them in parallel

None means 1 unless in a joblibparallelbackend context1means using all

processors See Glossary for more details

randomstate int RandomState instance or None default Used for randomizing the singular

value decomposition and the kmeans initialization Use an int to make the randomness

deterministic See Glossary

Attributes

rows arraylike shape nrowclusters nrow Results of the clustering rowsi r is

True if cluster iconains row r Available only after calling fit

64sklearnclusterbicluster Biclustering 1521

scikitlearn user guide Release 0213

columns arraylike shape ncolumnclusters ncolumns Results of the clustering like
rows

rowlabels arraylike shape nrows The bicluster label of each row

columnlabels arraylike shape ncols The bicluster label of each column

References

- Dhillon Inderjit S 2001 Coclustering documents and words using bipartite spectral graph partitioning

Examples

```
from sklearncluster import SpectralCoclustering
```

```
import numpy as np
```

```
X nparray1 1 2 1 1 0
```

```
4 7 3 5 3 6
```

```
clustering SpectralCoclusteringnclusters2 randomstate0fitX
```

```
clusteringrowlabels
```

```
array0 1 1 0 0 0 dtypeint32
```

```
clusteringcolumnlabels
```

```
array0 0 dtypeint32
```

```
clustering
```

```
SpectralCoclusteringinitkmeans minibatchFalse nclusters2
```

```
ninit10 njobsNone nsvdvecsNone randomstate0
```

```
svdmethodrandomized
```

Methods

fitself X y Creates a biclustering for X

getindices self i Row and column indices of the i'th bicluster

getparams self deep Get parameters for this estimator

getshape self i Shape of the i'th bicluster

getsubmatrix self i data Returns the submatrix corresponding to bicluster i

setparams self params Set the parameters of this estimator

init selfnclusters3 svdmethod'randomized' nsvdvecsNone minibatchFalse

init'kmeans' ninit10 njobsNone randomstateNone

biclusters

Convenient way to get row and column indicators together

Returns the rows andcolumns members

fitselfXyNone

Creates a biclustering for X

Parameters

Xarraylike shape nsamples nfeatures

ylgnored

getindices selfi

Row and column indices of the i'th bicluster

1522 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Only works if rows and columns attributes exist

Parameters

iint The index of the cluster

Returns

rowind nparray dtype npintp Indices of rows in the dataset that belong to the bicluster

colind nparray dtype npintp Indices of columns in the dataset that belong to the bicluster

ter

getparams selfdeep True

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getshape selfi

Shape of the i'th bicluster

Parameters

iint The index of the cluster

Returns

shape int int Number of rows and columns resp in the bicluster

getsubmatrix selfidata

Returns the submatrix corresponding to bicluster i

Parameters

iint The index of the cluster

data array The data

Returns

submatrix array The submatrix corresponding to bicluster i

Notes

Works with sparse matrices Only works if rows and columns attributes exist

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

64sklearnclusterbicluster Biclustering 1523

scikitlearn user guide Release 0213

65sklearncompose Composite Estimators

Metaestimators for building composite models with transformers

In addition to its current contents this module will eventually be home to refurbished versions of Pipeline and FeatureUnion

User guide See the Pipelines and composite estimators section for further details

composeColumnTransformer transformers Applies transformers to columns of an array or pandas DataFrame

composeTransformedTargetRegressor Metaestimator to regress on a transformed target

651sklearncompose ColumnTransformer

classsklearncompose ColumnTransformer transformers remainder'drop'

sparsethreshold03 njobsNone trans

formerweightsNone verboseFalse

Applies transformers to columns of an array or pandas DataFrame

This estimator allows different columns or column subsets of the input to be transformed separately and the features generated by each transformer will be concatenated to form a single feature space This is useful for heterogeneous or columnar data to combine several feature extraction mechanisms or transformations into a single transformer

Read more in the User Guide

New in version 020

Parameters

transformers list of tuples List of name transformer columns tuples specifying the transformer objects to be applied to subsets of the data

name string Like in Pipeline and FeatureUnion this allows the transformer and its parameters to be set using setparams and searched in grid search

transformer estimator or 'passthrough' 'drop' Estimator must support fit and

transform Specialcased strings 'drop' and 'passthrough' are accepted as well to

indicate to drop the columns or to pass them through untransformed respectively

columns string or int arraylike of string or int slice boolean mask array or callable

Indexes the data on its second axis Integers are interpreted as positional columns while

strings can reference DataFrame columns by name A scalar string or int should be used

wheretransformer expects X to be a 1d arraylike vector otherwise a 2d array will

be passed to the transformer A callable is passed the input data Xand can return any of the above

remainder 'drop' 'passthrough' or estimator default 'drop' By default only the

specified columns in transformers are transformed and combined in the output and

the nonspecified columns are dropped default of drop By specifying

remainderpassthrough all remaining columns that were not specified in trans

formers will be automatically passed through This subset of columns is concatenated with

the output of the transformers By setting remainder to be an estimator the remaining

nonspecified columns will use the remainder estimator The estimator must support fit

andtransform Note that using this feature requires that the DataFrame columns input at fit

andtransform have identical order

1524 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sparsethreshold float default 0.3 If the output of the different transformers contains sparse matrices these will be stacked as a sparse matrix if the overall density is lower than this value Usesparsethreshold0 to always return dense When the transformed output consists of all dense data the stacked result will be dense and this keyword will be ignored
njobs int or None optional defaultNone Number of jobs to run in parallel None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

transformerweights dict optional Multiplicative weights for features per transformer The output of the transformer is multiplied by these weights Keys are transformer names values the weights

verbose boolean optionaldefaultFalse If True the time elapsed while fitting each transformer will be printed as it is completed

Attributes

transformers list The collection of fitted transformers as tuples of name fittedtransformer columnfittedtransformer can be an estimator 'drop' or 'passthrough' In case there were no columns selected this will be the unfitted transformer If there are remaining columns the final element is a tuple of the form 'remainder' transformer remainingcolumns corresponding to the remainder parameter If there are remaining columns then lentransformerslentransformers1 other wiselentransformerslentransformers

namedtransformers Bunch object a dictionary with attribute access Access the fitted transformer by name

sparseoutput boolean Boolean flag indicating whether the output of transform is a sparse matrix or a dense numpy array which depends on the output of the individual transformers and thesparsethreshold keyword

See also

sklearncomposemakecolumntransformer convenience function for combining the outputs of multiple transformer objects applied to column subsets of the original feature space

Notes

The order of the columns in the transformed feature matrix follows the order of how the columns are specified in the transformers list Columns of the original feature matrix that are not specified are dropped from the resulting transformed feature matrix unless specified in the passthrough keyword Those columns specified withpassthrough are added at the right to the output of the transformers

Examples

```
import numpy as np
from sklearncompose import ColumnTransformer
from sklearnpreprocessing import Normalizer
ct = ColumnTransformer(
    norm1 Normalizer(norml1=0.1)
    norm2 Normalizer(norml1=2)
X nparray0 1 2 2
1 1 0 1
Normalizer scales each row of X to unit norm A separate scaling
```

scikitlearn user guide Release 0213

is applied for the two first and two last elements of each row independently

ctffittransformX

array0 1 05 05

05 05 0 1

Methods

fitself X y Fit all transformers using X

fittransform self X y Fit all transformers transform the data and concatenate results

getfeaturenames self Get feature names from all transformers

getparams self deep Get parameters for this estimator

setparams self kwargs Set the parameters of this estimator

transform self X Transform X separately by each transformer concatenate results

init selftransformers remainder'drop' sparsethreshold03 njobsNone transformerweightsNone verboseFalse

fitselfXyNone

Fit all transformers using X

Parameters

Xarraylike or DataFrame of shape nsamples nfeatures Input data of which specified subsets are used to fit the transformers

yarraylike shape nsamples optional Targets for supervised learning

Returns

self ColumnTransformer This estimator

fittransform selfXyNone

Fit all transformers transform the data and concatenate results

Parameters

Xarraylike or DataFrame of shape nsamples nfeatures Input data of which specified subsets are used to fit the transformers

yarraylike shape nsamples optional Targets for supervised learning

Returns

Xt arraylike or sparse matrix shape nsamples sumncomponents hstack of results of transformers sumncomponents is the sum of ncomponents output dimension over transformers If any result is a sparse matrix everything will be converted to sparse matrices

getfeaturenames self

Get feature names from all transformers

Returns

featurenames list of strings Names of the features produced by transform

1526 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

`getparams selfdeepTrue`
Get parameters for this estimator

Parameters

`deep` boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

`params` mapping of string to any Parameter names mapped to their values

`namedtransformers`

Access the fitted transformer by name

Readonly attribute to access any transformer by given name Keys are transformer names and values are the fitted transformer objects

`setparams selfkwargs`
Set the parameters of this estimator

Valid parameter keys can be listed with `getparams`

Returns

`self`

`transform selfX`
Transform X separately by each transformer concatenate results

Parameters

`X` arraylike or DataFrame of shape `nsamples nfeatures` The data to be transformed

`by` subset

Returns

`Xt` arraylike or sparse matrix shape `nsamples sumncomponents` hstack of results of transformers `sumncomponents` is the sum of `ncomponents` output dimension over transformers If any result is a sparse matrix everything will be converted to sparse matrices

Examples using `sklearncomposeColumnTransformer`

- Column Transformer with Mixed Types
- Column Transformer with Heterogeneous Data Sources

652sklearncompose TransformedTargetRegressor

`class sklearncompose TransformedTargetRegressor regressorNone transformerNone`

`funcNone inversefuncNone`

`checkinverseTrue`

Metaestimator to regress on a transformed target

Useful for applying a nonlinear transformation in regression problems This transformation can be given as a Transformer such as the `QuantileTransformer` or as a function and its inverse such as `log` and `exp`

The computation during fit is

`regressorfitX funcy`

65sklearncompose Composite Estimators 1527

scikitlearn user guide Release 0213

or
regressorfitX transformertransformy

The computation during predict is

inversefuncregressorpredictX

or

transformerinversetransformregressorpredictX

Read more in the User Guide

Parameters

regressor object defaultLinearRegression Regressor object such as derived from
RegressorMixin This regressor will automatically be cloned each time prior to fitting

transformer object defaultNone Estimator object such as derived from
TransformerMixin Cannot be set at the same time as func andinversefunc If

transformer isNone as well asfunc andinversefunc the transformer will be
an identity transformer Note that the transformer will be cloned during fitting Also the
transformer is restricting yto be a numpy array

func function optional Function to apply to ybefore passing to fit Cannot be set at the
same time as transformer The function needs to return a 2dimensional array If func
isNone the function used will be the identity function
inversefunc function optional Function to apply to the prediction of the regressor Can
not be set at the same time as transformer as well The function needs to return a
2dimensional array The inverse function is used to return predictions to the same space of
the original training labels

checkinverse bool defaultTrue Whether to check that transform followed by
inversetransform orfunc followed by inversefunc leads to the original tar
gets

Attributes

regressor object Fitted regressor

transformer object Transformer used in fit andpredict

Notes

Internally the target yis always converted into a 2dimensional array to be used by scikitlearn transformers

At the time of prediction the output will be reshaped to a have the same number of dimensions as y

Seexamplescomposeplottransformedtargetpy

Examples

```
import numpy as np
from sklearnlinearmodel import LinearRegression
from sklearncompose import TransformedTargetRegressor
tt TransformedTargetRegressorregressorLinearRegression
funcnplog inversefuncnpexp
```

1528 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

X nparange4reshape1 1

y npexp2 Xravel

ttfitX y

TransformedTargetRegressor

ttscoreX y

10

ttregressorcoef

array2

Methods

fitself X y sampleweight Fit the model according to the given training data

getparams self deep Get parameters for this estimator

predict self X Predict using the base regressor applying inverse

score self X y sampleweight Returns the coefficient of determination R2 of the pre
diction

setparams self params Set the parameters of this estimator

init self regressorNone transformerNone funcNone inversefuncNone

checkinverseTrue

fitselfXysampleweightNone

Fit the model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vector where
nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target values

sampleweight arraylike shape nsamples optional Array of weights that are assigned
to individual samples If not provided then each sample is given unit weight

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the base regressor applying inverse

The regressor is used to predict and the inversefunc orinversetransform is applied before
returning the prediction

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Samples

65sklearncompose Composite Estimators 1529

scikitlearn user guide Release 0213

Returns

yhat array shape nsamples Predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearncomposeTransformedTargetRegressor

•Effect of transforming the targets in regression model

composemakecolumntransformer Construct a ColumnTransformer from the given transform

ers

653sklearncompose makecolumntransformer

sklearncompose makecolumntransformer transformers kwargs

Construct a ColumnTransformer from the given transformers

1530 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

This is a shorthand for the ColumnTransformer constructor it does not require and does not permit naming the transformers Instead they will be given names automatically based on their types It also does not allow weighting with transformerweights

Parameters

transformers tuples of transformers and column selections

remainder 'drop' 'passthrough' or estimator default 'drop' By default only the specified columns in transformers are transformed and combined in the output and the nonspecified columns are dropped default of drop By specifying

remainderpassthrough all remaining columns that were not specified in transformers will be automatically passed through This subset of columns is concatenated with the output of the transformers By setting remainder to be an estimator the remaining nonspecified columns will use the remainder estimator The estimator must support fit andtransform

sparsethreshold float default 03 If the transformed output consists of a mix of sparse and dense data it will be stacked as a sparse matrix if the density is lower than this value Use sparsethreshold0 to always return dense When the transformed output consists of all sparse or all dense data the stacked result will be sparse or dense respectively and this keyword will be ignored

njobs int or None optional defaultNone Number of jobs to run in parallel None means 1 unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

verbose boolean optionaldefaultFalse If True the time elapsed while fitting each transformer will be printed as it is completed

Returns

ctColumnTransformer

See also

sklearncomposeColumnTransformer Class that allows combining the outputs of multiple transformer objects used on column subsets of the data into a single feature space

Examples

```
from sklearnpreprocessing import StandardScaler OneHotEncoder
from sklearncompose import makecolumntransformer
makecolumntransformer
StandardScaler numericalcolumn
OneHotEncoder categoricalcolumn
```

ColumnTransformernjobsNone remainderdrop sparsethreshold03

transformerweightsNone

transformersstandardscaler

StandardScaler

numericalcolumn

onehotencoder

OneHotEncoder

categoricalcolumn verboseFalse

65sklearncompose Composite Estimators 1531

scikitlearn user guide Release 0213

66sklearncovariance Covariance Estimators

The sklearncovariance module includes methods and algorithms to robustly estimate the covariance of features given a set of points. The precision matrix defined as the inverse of the covariance is also estimated. Covariance estimation is closely related to the theory of Gaussian Graphical Models. User guide See the Covariance estimation section for further details.

covarianceEmpiricalCovariance Maximum likelihood covariance estimator

covarianceEllipticEnvelope An object for detecting outliers in a Gaussian distributed dataset

covarianceGraphicalLasso alpha mode Sparse inverse covariance estimation with an l_1 penalized estimator

covarianceGraphicalLassoCV alphas Sparse inverse covariance w crossvalidated choice of the l_1 penalty

covarianceLedoitWolf storeprecision LedoitWolf Estimator

covarianceMinCovDet storeprecision Minimum Covariance Determinant MCD robust estimator of covariance

covarianceOAS storeprecision Oracle Approximating Shrinkage Estimator

covarianceShrunkCovariance Covariance estimator with shrinkage

661sklearncovariance EmpiricalCovariance

class sklearncovariance EmpiricalCovariance storeprecision True as sume centered False

Maximum likelihood covariance estimator

Read more in the User Guide

Parameters

storeprecision bool Specifies if the estimated precision is stored

assume centered bool If True data are not centered before computation. Useful when working with data whose mean is almost but not exactly zero. If False default data are centered before computation.

Attributes

location arraylike shape nfeatures Estimated location ie the estimated mean

covariance 2D ndarray shape nfeatures nfeatures Estimated covariance matrix

precision 2D ndarray shape nfeatures nfeatures Estimated pseudoinverse matrix stored only if storeprecision is True

Examples

```
import numpy as np
```

```
from sklearncovariance import EmpiricalCovariance
```

```
from sklearn.datasets import make_gaussian_quantiles
```

```
realcov = np.array(8, 3,
```

```
3, 4,
```

```
rng = np.random.RandomState(0)
```

```
X = rng.multivariate_normal(mean=0, 0,
```

1532 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

covrealcov
size500
cov EmpiricalCovariancefitX
covcovariance
array07569 02818
02818 03928
covlocation
array00622 00193

Methods
errornorm self compcov norm scaling Computes the Mean Squared Error between two covari
ance estimators
fitself X y Fits the Maximum Likelihood Estimator covariance
model according to the given training data and param
eters
getparams self deep Get parameters for this estimator
getprecision self Getter for the precision matrix
mahalanobis self X Computes the squared Mahalanobis distances of given
observations
score self Xtest y Computes the loglikelihood of a Gaussian data set with
selfcovariance as an estimator of its covari
ance matrix
setparams self params Set the parameters of this estimator
init selfstoreprecisionTrue assumecenteredFalse
errornorm selfcompcov norm'frobenius' scalingTrue squaredTrue
Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius
norm

Parameters
compcov arraylike shape nfeatures nfeatures The covariance to compare with
norm str The type of norm used to compute the error Available error types 'frobenius'
default sqrttrAtA 'spectral' sqrtmaxeigenvaluesAtA where A is the error
compcov selfcovariance
scaling bool If True default the squared error norm is divided by nfeatures If False
the squared error norm is not rescaled
squared bool Whether to compute the squared error norm or the error norm If True
default the squared error norm is returned If False the error norm is returned

Returns
The Mean Squared Error in the sense of the Frobenius norm between
self andcompcov covariance estimators
fitselfXyNone
Fits the Maximum Likelihood Estimator covariance model according to the given training data and param
eters
Parameters
66sklearn covariance Covariance Estimators 1533

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

ynot used present for API (Application Programming Interface) consistence purpose

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the likelihood where nsamples is the number of samples and nfeatures is the number of features Xtest is assumed to be drawn from the same distribution than the data used in fit including centering

ynot used present for API (Application Programming Interface) consistence purpose

Returns

resfloat The likelihood of the data set with selfcovariance as an estimator of its covariance matrix

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

1534 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns
self

Examples using sklearn.covariance.EmpiricalCovariance

- Robust covariance estimation and Mahalanobis distances relevance
- Robust vs Empirical covariance estimate

662sklearn.covariance.EllipticEnvelope

class sklearn.covariance.EllipticEnvelope store_precision=True assume_centered=False
support_fraction=None contamination=0.1
random_state=None

An object for detecting outliers in a Gaussian distributed dataset

Read more in the User Guide

Parameters

store_precision boolean optional default=True Specify if the estimated precision is stored

assume_centered boolean optional default=False If True the support of robust location and covariance estimates is computed and a covariance estimate is recomputed from it without centering the data Useful to work with data whose mean is significantly equal to zero but is not exactly zero If False the robust location and covariance are directly computed with the FastMCD algorithm without additional treatment

support_fraction float in [0, 1] optional default=None The proportion of points to be included in the support of the raw MCD estimate If None the minimum value of support_fraction will be used within the algorithm n_samples n_features 1

2

contamination float in [0, 0.5] optional default=0.1 The amount of contamination of the data set ie the proportion of outliers in the data set

random_state int RandomState instance or None optional default=None The seed of the pseudo random number generator to use when shuffling the data If int random_state is the seed used by the random number generator If RandomState instance random_state is the random number generator If None the random number generator is the RandomState instance used by np.random

Attributes

location arraylike shape n_features Estimated robust location

covariance arraylike shape n_features n_features Estimated robust covariance matrix

precision arraylike shape n_features n_features Estimated pseudo inverse matrix stored only if store_precision is True

support arraylike shape n_samples A mask of the observations that have been used to compute the robust estimates of location and shape

offset float Offset used to define the decision function from the raw scores We have the relation decision_function scoresamples_offset The offset depends on the contamination parameter and is defined in such a way we obtain the expected number of outliers samples with decision function 0 in training

See also

66sklearn.covariance.CovarianceEstimators 1535

scikitlearn user guide Release 0213

EmpiricalCovariance MinCovDet

Notes

Outlier detection from covariance estimation may break or not perform well in highdimensional settings In particular one will always take care to work with nsamples nfeatures 2

References

R68ae096da0e41

Examples

```
import numpy as np
from sklearn.covariance import EllipticEnvelope
truecov = np.array([
    [3, 4],
    [4, 3]
])
X = np.random.RandomState(0).multivariate_normal(mean=[0, 0],
    cov=truecov,
    size=500)
cov = EllipticEnvelope().fit(X)
predict returns 1 for an inlier and 0 for an outlier
cov.predict([
    [0, 0],
    [1, 1]
])
array([1, 1])
cov.covariance_
array([[0.7411, 0.2535],
       [0.2535, 0.3053]])
cov.location_
array([0.0813, 0.0427])
```

Methods

correctcovariance(self, data) Apply a correction to raw Minimum Covariance Determinant estimates

decisionfunction(self, X) Compute the decision function of the given observations

errornorm(self, compcov, norm, scaling) Computes the Mean Squared Error between two covariance estimators

fit(self, X, y) Fit the EllipticEnvelope model

fitpredict(self, X, y) Performs fit on X and returns labels for X

getparams(self, deep) Get parameters for this estimator

getprecision(self) Getter for the precision matrix

mahalanobis(self, X) Computes the squared Mahalanobis distances of given observations

predict(self, X) Predict the labels 1 inlier 0 outlier of X according to the fitted model

reweightcovariance(self, data) Reweight raw Minimum Covariance Determinant estimates

Continued on next page

scikitlearn user guide Release 0213

Table 634 – continued from previous page

score self X y sampleweight Returns the mean accuracy on the given test data and labels

scoresamples self X Compute the negative Mahalanobis distances

setparams self params Set the parameters of this estimator

init selfstoreprecisionTrue assumecenteredFalse supportfractionNone contamination01 randomstateNone

correctcovariance selfdata

Apply a correction to raw Minimum Covariance Determinant estimates

Correction using the empirical correction factor suggested by Rousseeuw and Van Driessen in RVD

Parameters

data arraylike shape nsamples nfeatures The data matrix with p features and n sam

ples The data set must be the one which was used to compute the raw estimates

Returns

covariancecorrected arraylike shape nfeatures nfeatures Corrected robust covariance estimate

References

RVD

decisionfunction selfXrawvaluesNone

Compute the decision function of the given observations

Parameters

Xarraylike shape nsamples nfeatures

rawvalues bool optional Whether or not to consider raw Mahalanobis distances as the

decision function Must be False default for compatibility with the others outlier detection tools

Deprecated since version 020 rawvalues has been deprecated in 020 and will be removed in 022

Returns

decision arraylike shape nsamples Decision function of the samples It is equal to

the shifted Mahalanobis distances The threshold for being an outlier is 0 which ensures a compatibility with other outlier detection algorithms

errornorm selfcompcov norm'frobenius' scalingTrue squaredTrue

Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius norm

Parameters

compcov arraylike shape nfeatures nfeatures The covariance to compare with

norm str The type of norm used to compute the error Available error types 'frobenius'

default sqrttrAtA 'spectral' sqrtmaxeigenvaluesAtA where A is the error

compcov selfcovariance

scaling bool If True default the squared error norm is divided by nfeatures If False the squared error norm is not rescaled

66sklearn covariance Covariance Estimators 1537

scikitlearn user guide Release 0213

squared bool Whether to compute the squared error norm or the error norm If True default the squared error norm is returned If False the error norm is returned

Returns

The Mean Squared Error in the sense of the Frobenius norm between self and compcov covariance estimators

fitselfXyNone

Fit the EllipticEnvelope model

Parameters

Xnumpy array or sparse matrix shape nsamples nfeatures Training data

yignored not used present for API (Application Programming Interface) consistency by convention

fitpredict selfXyNone

Performs fit on X and returns labels for X

Returns 1 for outliers and 1 for inliers

Parameters

Xndarray shape nsamples nfeatures Input data

yignored not used present for API (Application Programming Interface) consistency by convention

Returns

yndarray shape nsamples 1 for inliers 1 for outliers

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

predictselfX

Predict the labels 1 inlier 1 outlier of X according to the fitted model

1538 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures

Returns

isinlier array shape nsamples Returns 1 for anomaliesoutliers and 1 for inliers

reweightcovariance selfdata

Reweight raw Minimum Covariance Determinant estimates

Reweight observations using Rousseeuw’s method equivalent to deleting outlying observations from the data set before computing location and covariance estimates described in RVDriessen

Parameters

data arraylike shape nsamples nfeatures The data matrix with p features and n sam

ples The data set must be the one which was used to compute the raw estimates

Returns

locationreweighted arraylike shape nfeatures Reweighted robust location esti
mate

coviancereweighted arraylike shape nfeatures nfeatures Reweighted robust co
variance estimate

supportreweighted arraylike type boolean shape nsamples A mask of the obser
vations that have been used to compute the reweighted robust location and covariance
estimates

References

RVDriessen

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each
sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

scoresamples selfX

Compute the negative Mahalanobis distances

Parameters

Xarraylike shape nsamples nfeatures

Returns

negativemahaldistances arraylike shape nsamples Opposite of the Mahalanobis
distances

66sklearn covariance Covariance Estimators 1539

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearncovarianceEllipticEnvelope

•Comparing anomaly detection algorithms for outlier detection on toy datasets

•Outlier detection on a real data set

663sklearncovariance GraphicalLasso

classsklearncovariance GraphicalLasso alpha001 mode'cd' tol00001

enettol00001 maxiter100 verboseFalse

assumecenteredFalse

Sparse inverse covariance estimation with an l1penalized estimator

Read more in the User Guide

Parameters

alpha positive float default 001 The regularization parameter the higher alpha the more regularization the sparser the inverse covariance

mode 'cd' 'lars' default 'cd' The Lasso solver to use coordinate descent or LARS Use

LARS for very sparse underlying graphs where p n Elsewhere prefer cd which is more numerically stable

tolpositive float default 1e4 The tolerance to declare convergence if the dual gap goes below this value iterations are stopped

enettol positive float optional The tolerance for the elastic net solver used to calculate the descent direction This parameter controls the accuracy of the search direction for a given column update not of the overall parameter estimate Only used for mode'cd'

maxiter integer default 100 The maximum number of iterations

verbose boolean default False If verbose is True the objective function and dual gap are plotted at each iteration

assumecentered boolean default False If True data are not centered before computation Useful when working with data whose mean is almost but not exactly zero If False data are centered before computation

Attributes

location arraylike shape nfeatures Estimated location ie the estimated mean

covariance arraylike shape nfeatures nfeatures Estimated covariance matrix

precision arraylike shape nfeatures nfeatures Estimated pseudo inverse matrix

niter int Number of iterations run

See also

1540 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

graphicallasso GraphicalLassoCV

Examples

```
import numpy as np
from sklearn.covariance import GraphicalLasso
truecov = np.array([[0.8, 0.0, 0.2, 0.0],
                    [0.0, 0.4, 0.0, 0.0],
                    [0.2, 0.0, 0.3, 0.1],
                    [0.0, 0.0, 0.1, 0.7]])
np.random.seed(0)
X = np.random.multivariate_normal(mean=[0, 0, 0, 0],
                                  cov=truecov,
                                  size=200)
cov = GraphicalLasso.fit(X,
```

```
np.array([[0.816, 0.049, 0.218, 0.019],
         [0.049, 0.364, 0.017, 0.034],
         [0.218, 0.017, 0.322, 0.093],
         [0.019, 0.034, 0.093, 0.69]]),
np.array([0.73, 0.04, 0.038, 0.143]),
decimals=3,
location=decimals=3)
```

```
array([[0.816, 0.049, 0.218, 0.019],
       [0.049, 0.364, 0.017, 0.034],
       [0.218, 0.017, 0.322, 0.093],
       [0.019, 0.034, 0.093, 0.69]])
array([0.73, 0.04, 0.038, 0.143])
```

Methods

errornorm self compcov norm scaling Computes the Mean Squared Error between two covariance estimators

fitself X y Fits the GraphicalLasso model to X

getparams self deep Get parameters for this estimator

getprecision self Getter for the precision matrix

mahalanobis self X Computes the squared Mahalanobis distances of given observations

score self Xtest y Computes the loglikelihood of a Gaussian data set with self.covariance as an estimator of its covariance matrix

setparams self params Set the parameters of this estimator

init self alpha=0.01 mode='cd' tol=0.0001 enet_tol=0.0001 max_iter=100 verbose=False

assume_centered=False

errornorm self compcov norm='frobenius' scaling=True squared=True

Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius norm

Parameters

compcov arraylike shape (n_features, n_features) The covariance to compare with

norm str The type of norm used to compute the error Available error types 'frobenius'

default sqrt(tr(A^T A)) 'spectral' sqrt(max(eigenvalues(A^T A))) where A is the error

compcov selfcovariance

scaling bool If True default the squared error norm is divided by n_features If False the squared error norm is not rescaled

scikitlearn user guide Release 0213

squared bool Whether to compute the squared error norm or the error norm If True default the squared error norm is returned If False the error norm is returned

Returns

The Mean Squared Error in the sense of the Frobenius norm between self andcompcov covariance estimators

fitselfXyNone

Fits the GraphicalLasso model to X

Parameters

Xndarray shape nsamples nfeatures Data from which to compute the covariance estimate

yignored

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the likelihood where nsamples is the number of samples and nfeatures is the number of features Xtest is assumed to be drawn from the same distribution than the data used in fit including centering

ynot used present for API (Application Programming Interface) consistence purpose

Returns

1542 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

resfloat The likelihood of the data set with selfcovariance as an estimator of its covariance matrix

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

664sklearn covariance GraphicalLassoCV

class sklearn covariance GraphicalLassoCV alphas4 nrefinements4 cv'warn'

tol00001 enettol00001 maxiter100

mode'cd' njobsNone verboseFalse as

sumecenteredFalse

Sparse inverse covariance w crossvalidated choice of the l1 penalty

See glossary entry for crossvalidation estimator

Read more in the User Guide

Parameters

alphas integer or list positive float optional If an integer is given it fixes the number of points

on the grids of alpha to be used If a list is given it gives the grid to be used See the notes

in the class docstring for more details

nrefinements strictly positive integer The number of times the grid is refined Not used if

explicit values of alphas are passed

cvint crossvalidation generator or an iterable optional Determines the crossvalidation split

ting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation
- integer to specify the number of folds
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in

v022

tolpositive float optional The tolerance to declare convergence if the dual gap goes below

this value iterations are stopped

enettol positive float optional The tolerance for the elastic net solver used to calculate the

descent direction This parameter controls the accuracy of the search direction for a given

column update not of the overall parameter estimate Only used for mode'cd'

maxiter integer optional Maximum number of iterations

66sklearn covariance Covariance Estimators 1543

scikitlearn user guide Release 0213

mode 'cd' 'lars' The Lasso solver to use coordinate descent or LARS Use LARS for
very sparse underlying graphs where number of features is greater than number of samples
Elsewhere prefer cd which is more numerically stable
njobs int or None optional defaultNone number of jobs to run in parallel None means 1
unless in a joblibparallelbackend context1means using all processors See
Glossary for more details

verbose boolean optional If verbose is True the objective function and duality gap are printed
at each iteration
assumecentered boolean If True data are not centered before computation Useful when
working with data whose mean is almost but not exactly zero If False data are centered
before computation

Attributes
location arraylike shape nfeatures Estimated location ie the estimated mean
covariance numpyndarray shape nfeatures nfeatures Estimated covariance matrix
precision numpyndarray shape nfeatures nfeatures Estimated precision matrix inverse
covariance
alpha float Penalization parameter selected
cvalphas list of float All penalization parameters explored
gridscores 2D numpyndarray nalphas nfolds Loglikelihood score on leftout data
across folds
niter int Number of iterations run for the optimal alpha

See also
graphicallasso GraphicalLasso
Notes
The search for the optimal penalization parameter alpha is done on an iteratively refined grid first the cross
validated scores on a grid are computed then a new refined grid is centered around the maximum and so on
One of the challenges which is faced here is that the solvers can fail to converge to a wellconditioned estimate
The corresponding values of alpha then come out as missing values but the optimum may be close to these
missing values

Examples
import numpy as np
from sklearn.covariance import GraphicalLassoCV
truecov nparray08 00 02 00
00 04 00 00
02 00 03 01
00 00 01 07
nprandomseed0
X nprandommultivariatenormalmean0 0 0 0
covtruecov
size200
cov GraphicalLassoCVcv5fitX

scikitlearn user guide Release 0213
nparoundcovcovariance decimals3
array0816 0051 022 0017
0051 0364 0018 0036
022 0018 0322 0094
0017 0036 0094 069
nparoundcovlocation decimals3
array0073 004 0038 0143
Methods
errornorm self compcov norm scaling Computes the Mean Squared Error between two covariance estimators
fitself X y Fits the GraphicalLasso covariance model to X
getparams self deep Get parameters for this estimator
getprecision self Getter for the precision matrix
mahalanobis self X Computes the squared Mahalanobis distances of given observations
score self Xtest y Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix
setparams self params Set the parameters of this estimator
init selfalphas4 nrefinements4 cv'warn' tol00001 enettol00001 maxiter100
mode'cd' njobsNone verboseFalse assumecenteredFalse
errornorm selfcompcov norm'frobenius' scalingTrue squaredTrue
Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius norm
Parameters
compcov arraylike shape nfeatures nfeatures The covariance to compare with
norm str The type of norm used to compute the error Available error types 'frobenius'
default sqrttrAtA 'spectral' sqrtmaxeigenvaluesAtA where A is the error
compcov selfcovariance
scaling bool If True default the squared error norm is divided by nfeatures If False the squared error norm is not rescaled
squared bool Whether to compute the squared error norm or the error norm If True default the squared error norm is returned If False the error norm is returned
Returns
The Mean Squared Error in the sense of the Frobenius norm between self andcompcov covariance estimators
fitselfXyNone
Fits the GraphicalLasso covariance model to X
Parameters
Xndarray shape nsamples nfeatures Data from which to compute the covariance estimate
yignored
66sklearn covariance Covariance Estimators 1545

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the likelihood where nsamples is the number of samples and nfeatures is the number of features Xtest is assumed to be drawn from the same distribution than the data used in fit including centering

y not used present for API (Application Programming Interface) consistence purpose

Returns

resfloat The likelihood of the data set with selfcovariance as an estimator of its covariance matrix

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearn covariance Graphical Lasso CV

•Visualizing the stock market structure

1546 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

•Sparse inverse covariance estimation

665sklearn covariance LedoitWolf

class sklearn covariance LedoitWolf storeprecision True assumecentered False

blocksize 1000

LedoitWolf Estimator

LedoitWolf is a particular form of shrinkage where the shrinkage coefficient is computed using O Ledoit and M Wolf’s formula as described in “A WellConditioned Estimator for LargeDimensional Covariance Matrices”

Ledoit and Wolf Journal of Multivariate Analysis V olume 88 Issue 2 February 2004 pages 365411

Read more in the User Guide

Parameters

storeprecision bool default True Specify if the estimated precision is stored

assumecentered bool default False If True data will not be centered before computation

Useful when working with data whose mean is almost but not exactly zero If False de

fault data will be centered before computation

blocksize int default 1000 Size of the blocks into which the covariance matrix will be split

during its LedoitWolf estimation This is purely a memory optimization and does not affect

results

Attributes

location arraylike shape nfeatures Estimated location ie the estimated mean

covariance arraylike shape nfeatures nfeatures Estimated covariance matrix

precision arraylike shape nfeatures nfeatures Estimated pseudo inverse matrix stored

only if storeprecision is True

shrinkage float 0 shrinkage 1 Coefficient in the convex combination used for the

computation of the shrunk estimate

Notes

The regularised covariance is

$$\frac{1}{n} \text{shrinkage} \text{cov} + \frac{1 - \text{shrinkage}}{n} \text{np.identity}(n_{\text{features}})$$

where $\mu = \text{trace}(\text{cov}) / n_{\text{features}}$ and shrinkage is given by the Ledoit and Wolf formula see References

References

“A WellConditioned Estimator for LargeDimensional Covariance Matrices” Ledoit and Wolf Journal of Mul

tivariate Analysis V olume 88 Issue 2 February 2004 pages 365411

Examples

import numpy as np

from sklearn covariance import LedoitWolf

realcov = np.array([2

2 8

66sklearn covariance Covariance Estimators 1547

scikitlearn user guide Release 0213
nprandomseed0
X nprandommultivariatenormalmean0 0
covrealcov
size50
cov LedoitWolf fitX
covcovariance
array04406 01616
01616 08022
covlocation
array 00595 00075
Methods
errornorm self compcov norm scaling Computes the Mean Squared Error between two covariance estimators
fitself X y Fits the LedoitWolf shrunk covariance model according to the given training data and parameters
getparams self deep Get parameters for this estimator
getprecision self Getter for the precision matrix
mahalanobis self X Computes the squared Mahalanobis distances of given observations
score self Xtest y Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix
setparams self params Set the parameters of this estimator
init selfstoreprecisionTrue assumecenteredFalse blocksize1000
errornorm selfcompcov norm'frobenius' scalingTrue squaredTrue
Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius norm
Parameters
compcov arraylike shape nfeatures nfeatures The covariance to compare with
norm str The type of norm used to compute the error Available error types 'frobenius' default sqrttrAtA 'spectral' sqrtmaxeigenvaluesAtA where A is the error
compcov selfcovariance
scaling bool If True default the squared error norm is divided by nfeatures If False the squared error norm is not rescaled
squared bool Whether to compute the squared error norm or the error norm If True default the squared error norm is returned If False the error norm is returned
Returns
The Mean Squared Error in the sense of the Frobenius norm between self andcompcov covariance estimators
fitselfXyNone
Fits the LedoitWolf shrunk covariance model according to the given training data and parameters
Parameters
1548 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

ynot used present for API (Application Programming Interface) consistence purpose

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the likelihood where nsamples is the number of samples and nfeatures is the number of features Xtest is assumed to be drawn from the same distribution than the data used in fit including centering

ynot used present for API (Application Programming Interface) consistence purpose

Returns

resfloat The likelihood of the data set with selfcovariance as an estimator of its covariance matrix

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

66sklearncovariance Covariance Estimators 1549

scikitlearn user guide Release 0213

Returns
self

Examples using sklearn.covariance.LedoitWolf

- LedoitWolf vs OAS estimation
- Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood
- Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA

666sklearn.covariance.MinCovDet
class sklearn.covariance.MinCovDet store_precision=True assume_centered=False
sup

port_fraction=None random_state=None
Minimum Covariance Determinant MCD robust estimator of covariance

The Minimum Covariance Determinant covariance estimator is to be applied on Gaussian distributed data but could still be relevant on data drawn from a unimodal symmetric distribution. It is not meant to be used with multimodal data: the algorithm used to fit a MinCovDet object is likely to fail in such a case. One should consider projection pursuit methods to deal with multimodal datasets.

Read more in the User Guide

Parameters

store_precision bool Specify if the estimated precision is stored
assume_centered bool If True the support of the robust location and the covariance estimates is computed and a covariance estimate is recomputed from it without centering the data. Useful to work with data whose mean is significantly equal to zero but is not exactly zero. If False the robust location and covariance are directly computed with the FastMCD algorithm without additional treatment.
support_fraction float 0 support_fraction 1 The proportion of points to be included in the support of the raw MCD estimate. Default is None which implies that the minimum value of support_fraction will be used within the algorithm. n_samples n_features 1 2
random_state int RandomState instance or None optional default=None If int random_state is the seed used by the random number generator. If RandomState instance random_state is the random number generator. If None the random number generator is the RandomState instance used by np.random.

Attributes

raw_location arraylike shape n_features The raw robust estimated location before correction and reweighting.
raw_covariance arraylike shape n_features n_features The raw robust estimated covariance before correction and reweighting.
raw_support arraylike shape n_samples A mask of the observations that have been used to compute the raw robust estimates of location and shape before correction and reweighting.
location arraylike shape n_features Estimated robust location
covariance arraylike shape n_features n_features Estimated robust covariance matrix

scikitlearn user guide Release 0213

precision arraylike shape nfeatures Estimated pseudo inverse matrix stored only if storeprecision is True

support arraylike shape nsamples A mask of the observations that have been used to compute the robust estimates of location and shape

dist arraylike shape nsamples Mahalanobis distances of the training set on which fit is called observations

References

R9f63e655f7bdRouseeuw1984 R9f63e655f7bdRousseeuw R9f63e655f7bdButlerDavies

Examples

```
import numpy as np
from sklearn.covariance import MinCovDet
from sklearn.datasets import make_gaussian_quantiles
```

```
realcov = np.array(8, 3
```

```
3, 4
```

```
rng = np.random.RandomState(0)
```

```
X = rng.multivariate_normal(mean=0, 0
```

```
cov=realcov
```

```
size=500
```

```
cov = MinCovDet(random_state=0).fit(X
```

```
cov.covariance
```

```
array(0.7411, 0.2535
```

```
0.2535, 0.3053
```

```
cov.location
```

```
array(0.0813, 0.0427
```

Methods

correct_covariance(self, data) Apply a correction to raw Minimum Covariance Deter

minant estimates

error_norm(self, compcov, norm, scaling) Computes the Mean Squared Error between two covari

ance estimators

fit(self, X, y) Fits a Minimum Covariance Determinant with the

FastMCD algorithm

get_params(self, deep) Get parameters for this estimator

get_precision(self) Getter for the precision matrix

mahalanobis(self, X) Computes the squared Mahalanobis distances of given

observations

reweight_covariance(self, data) Reweight raw Minimum Covariance Determinant esti

mates

score(self, Xtest, y) Computes the loglikelihood of a Gaussian data set with

self.covariance as an estimator of its covari

ance matrix

set_params(self, params) Set the parameters of this estimator

66sklearn.covariance Covariance Estimators 1551

scikitlearn user guide Release 0213

init selfstoreprecisionTrue assumecenteredFalse supportfractionNone ran
domstateNone

correctcovariance selfdata

Apply a correction to raw Minimum Covariance Determinant estimates
Correction using the empirical correction factor suggested by Rousseeuw and Van Driessen in RVD

Parameters

data arraylike shape nsamples nfeatures The data matrix with p features and n sam
ples The data set must be the one which was used to compute the raw estimates

Returns

covariancecorrected arraylike shape nfeatures nfeatures Corrected robust covari
ance estimate

References

RVD

errornorm selfcompcov norm'frobenius' scalingTrue squaredTrue

Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius
norm

Parameters

compcov arraylike shape nfeatures nfeatures The covariance to compare with
norm str The type of norm used to compute the error Available error types 'frobenius'
default sqrttrAtA 'spectral' sqrtmaxeigenvaluesAtA where A is the error

compcov selfcovariance

scaling bool If True default the squared error norm is divided by nfeatures If False
the squared error norm is not rescaled

squared bool Whether to compute the squared error norm or the error norm If True
default the squared error norm is returned If False the error norm is returned

Returns

The Mean Squared Error in the sense of the Frobenius norm between
self andcompcov covariance estimators

fitselfXyNone

Fits a Minimum Covariance Determinant with the FastMCD algorithm

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the num
ber of samples and nfeatures is the number of features

ynot used present for API (Application Programming Interface) consistence purpose

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

1552 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

reweightcovariance selfdata

Reweight raw Minimum Covariance Determinant estimates

Reweight observations using Rousseeuw’s method equivalent to deleting outlying observations from the data set before computing location and covariance estimates described in RVDriessen

Parameters

data arraylike shape nsamples nfeatures The data matrix with p features and n samples The data set must be the one which was used to compute the raw estimates

Returns

locationreweighted arraylike shape nfeatures Reweighted robust location estimate

coviancereweighted arraylike shape nfeatures nfeatures Reweighted robust covariance estimate

supportreweighted arraylike type boolean shape nsamples A mask of the observations that have been used to compute the reweighted robust location and covariance estimates

References

RVDriessen

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the likelihood where nsamples is the number of samples and nfeatures is the number of

66sklearn covariance Covariance Estimators 1553

scikitlearn user guide Release 0213

features Xtest is assumed to be drawn from the same distribution than the data used in fit including centering

ynot used present for API (Application Programming Interface) consistence purpose

Returns

resfloat The likelihood of the data set with selfcovariance as an estimator of its covariance matrix

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearncovarianceMinCovDet

- Robust covariance estimation and Mahalanobis distances relevance
- Robust vs Empirical covariance estimate

667sklearncovariance OAS

classsklearncovariance OASstoreprecisionTrue assumecenteredFalse

Oracle Approximating Shrinkage Estimator

Read more in the User Guide

OAS is a particular form of shrinkage described in “Shrinkage Algorithms for MMSE Covariance Estimation” Chen et al IEEE Trans on Sign Proc V olume 58 Issue 10 October 2010

The formula used here does not correspond to the one given in the article In the original article formula 23 states that $2p$ is multiplied by Tracecovcov in both the numerator and denominator but this operation is omitted because for a large p the value of $2p$ is so small that it doesn't affect the value of the estimator

Parameters

storeprecision bool defaultTrue Specify if the estimated precision is stored

assumecentered bool defaultFalse If True data will not be centered before computation

Useful when working with data whose mean is almost but not exactly zero If False default data will be centered before computation

Attributes

covariance arraylike shape nfeatures nfeatures Estimated covariance matrix

precision arraylike shape nfeatures nfeatures Estimated pseudo inverse matrix stored only if storeprecision is True

shrinkage float 0 shrinkage 1 coefficient in the convex combination used for the computation of the shrunk estimate

1554 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Notes

The regularised covariance is

$\frac{1}{n} \text{shrinkage} \text{cov} \text{shrinkage} \mu \text{npidentitynfeatures}$

where $\mu = \text{tracecov} / \text{nfeatures}$ and shrinkage is given by the OAS formula see References

References

“Shrinkage Algorithms for MMSE Covariance Estimation” Chen et al IEEE Trans on Sign Proc Volume 58 Issue 10 October 2010

Methods

`errornorm self compcov norm scaling` Computes the Mean Squared Error between two covariance estimators

`fitself X y` Fits the Oracle Approximating Shrinkage covariance

`model according to the given training data and parameters`

`getparams self deep` Get parameters for this estimator

`getprecision self` Getter for the precision matrix

`mahalanobis self X` Computes the squared Mahalanobis distances of given observations

`score self Xtest y` Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

`setparams self params` Set the parameters of this estimator

`init selfstoreprecisionTrue assumecenteredFalse`

`errornorm self compcov norm ‘frobenius’ scalingTrue squaredTrue`

Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius norm

Parameters

`compcov arraylike shape nfeatures nfeatures` The covariance to compare with

`norm str` The type of norm used to compute the error Available error types ‘frobenius’

`default sqrttrAtA ‘spectral’ sqrtmaxeigenvaluesAtA` where A is the error

`compcov selfcovariance`

`scaling bool` If True default the squared error norm is divided by nfeatures If False the squared error norm is not rescaled

`squared bool` Whether to compute the squared error norm or the error norm If True default the squared error norm is returned If False the error norm is returned

Returns

The Mean Squared Error in the sense of the Frobenius norm between

self andcompcov covariance estimators

66sklearn covariance Covariance Estimators 1555

scikitlearn user guide Release 0213

fitselfXyNone

Fits the Oracle Approximating Shrinkage covariance model according to the given training data and parameters

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

ynot used present for API (Application Programming Interface) consistence purpose

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the likelihood where nsamples is the number of samples and nfeatures is the number of features Xtest is assumed to be drawn from the same distribution than the data used in fit including centering

ynot used present for API (Application Programming Interface) consistence purpose

Returns

resfloat The likelihood of the data set with selfcovariance as an estimator of its covariance matrix

1556 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

Examples using sklearncovarianceOAS

•LedoitWolf vs OAS estimation

•Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood

668sklearncovariance ShrunkCovariance

classsklearncovariance ShrunkCovariance storeprecisionTrue assumecenteredFalse

shrinkage01

Covariance estimator with shrinkage

Read more in the User Guide

Parameters

storeprecision boolean default True Specify if the estimated precision is stored

assumecentered boolean default False If True data will not be centered before computation Useful when working with data whose mean is almost but not exactly zero If False data will be centered before computation

shrinkage float 0 shrinkage 1 default 01 Coefficient in the convex combination used for the computation of the shrunk estimate

Attributes

location arraylike shape nfeatures Estimated location ie the estimated mean

covariance arraylike shape nfeatures nfeatures Estimated covariance matrix

precision arraylike shape nfeatures nfeatures Estimated pseudo inverse matrix stored only if storeprecision is True

shrinkage float 0 shrinkage 1 Coefficient in the convex combination used for the computation of the shrunk estimate

Notes

The regularized covariance is given by

$\frac{1}{n} \text{shrinkage} \cdot \text{cov} + \frac{1 - \text{shrinkage}}{n} \text{trace}(\text{cov}) \cdot \text{I}$ where $\text{mu} = \frac{1}{n} \sum \text{features}$

66sklearncovariance Covariance Estimators 1557

Examples

```
import numpy as np
from sklearn.covariance import ShrunkCovariance
from sklearn.datasets import make_gaussian_quantiles
realcov = np.array(8, 3, 4)
rng = np.random.RandomState(0)
X = rng.multivariate_normal(mean=[0, 0],
                             cov=realcov,
                             size=500)
cov = ShrunkCovariance().fit(X)
cov.covariance_
array([[0.7387, 0.2536, 0.2536, 0.411],
       [0.2536, 0.411, 0.0622, 0.0193],
       [0.0622, 0.0193, 0.0193, 0.0193],
       [0.0193, 0.0193, 0.0193, 0.0193]])
```

Methods

```
error_norm(self, compcov, norm='frobenius', scaling=True, squared=True)
    Computes the Mean Squared Error between two covariance estimators
    In the sense of the Frobenius norm

fit(self, X, y)
    Fits the shrunk covariance model according to the given training data
    and parameters

get_params(self, deep=True)
    Get parameters for this estimator

get_precision(self)
    Getter for the precision matrix

mahalanobis(self, X)
    Computes the squared Mahalanobis distances of given observations

score(self, Xtest, y)
    Computes the loglikelihood of a Gaussian data set with self.covariance_
    as an estimator of its covariance matrix

set_params(self, **params)
    Set the parameters of this estimator

init(self, store_precision=True, assume_centered=False, shrinkage=0.1)
    Initialize the estimator

error_norm(self, compcov, norm='frobenius', scaling=True, squared=True)
    Computes the Mean Squared Error between two covariance estimators
    In the sense of the Frobenius norm
```

Parameters

```
compcov : array-like, shape (n_features, n_features)
    The covariance to compare with

norm : str
    The type of norm used to compute the error. Available error types:
    'frobenius' (default), 'spectral', 'sqrt_max_eigenvalues'.
    A where A is the error matrix.

scaling : bool
    If True (default), the squared error norm is divided by n_features.
    If False, the squared error norm is not rescaled.

squared : bool
    Whether to compute the squared error norm or the error norm.
    If True (default), the squared error norm is returned.
    If False, the error norm is returned.
```

Returns

The Mean Squared Error in the sense of the Frobenius norm between
1558 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

self andcompcov covariance estimators

fitselfXyNone

Fits the shrunk covariance model according to the given training data and parameters

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

ynot used present for API (Application Programming Interface) consistence purpose

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the likelihood where nsamples is the number of samples and nfeatures is the number of features Xtest is assumed to be drawn from the same distribution than the data used in fit including centering

ynot used present for API (Application Programming Interface) consistence purpose

Returns

resfloat The likelihood of the data set with selfcovariance as an estimator of its covariance matrix

66sklearncomvariance Covariance Estimators 1559

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearncovarianceShrunkCovariance

- Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood
 - Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA
- covarianceempiricalcovariance X Computes the Maximum likelihood covariance estimator
covariancegraphicallasso empcov alpha
l1penalized covariance estimator

covarianceledoitwolf X assumecentered

Estimates the shrunk LedoitWolf covariance matrix

covarianceeoas X assumecentered Estimate covariance with the Oracle Approximating

Shrinkage algorithm

covarianceshrunkcovariance empcov Calculates a covariance matrix shrunk on the diagonal

669sklearncovariance empiricalcovariance

sklearncovariance empiricalcovariance XassumecenteredFalse

Computes the Maximum likelihood covariance estimator

Parameters

Xndarray shape nsamples nfeatures Data from which to compute the covariance estimate
assumecentered boolean If True data will not be centered before computation Useful when working with data whose mean is almost but not exactly zero If False data will be centered before computation

Returns

covariance 2D ndarray shape nfeatures nfeatures Empirical covariance Maximum Likelihood Estimator

Examples using sklearncovarianceempiricalcovariance

- Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood

6610sklearncovariance graphicallasso

sklearncovariance graphicallasso empcov alpha covinitNone mode'cd' tol00001

enettol00001 maxiter100 verboseFalse re

turncostsFalse eps2220446049250313e16 re

turnniterFalse

l1penalized covariance estimator

1560 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

empcov 2D ndarray shape nfeatures nfeatures Empirical covariance from which to compute the covariance estimate
alpha positive float The regularization parameter the higher alpha the more regularization the sparser the inverse covariance
covinit 2D array nfeatures nfeatures optional The initial guess for the covariance
mode 'cd' 'lars' The Lasso solver to use coordinate descent or LARS Use LARS for very sparse underlying graphs where $p \gg n$ Elsewhere prefer cd which is more numerically stable
tolpositive float optional The tolerance to declare convergence if the dual gap goes below this value iterations are stopped
enettol positive float optional The tolerance for the elastic net solver used to calculate the descent direction This parameter controls the accuracy of the search direction for a given column update not of the overall parameter estimate Only used for mode 'cd'
maxiter integer optional The maximum number of iterations
verbose boolean optional If verbose is True the objective function and dual gap are printed at each iteration
returncosts boolean optional If returncosts is True the objective function and dual gap at each iteration are returned
eps float optional The machineprecision regularization in the computation of the Cholesky diagonal factors Increase this for very illconditioned systems
returnniter bool optional Whether or not to return the number of iterations

Returns

covariance 2D ndarray shape nfeatures nfeatures The estimated covariance matrix
precision 2D ndarray shape nfeatures nfeatures The estimated sparse precision matrix
costs list of objective dualgap pairs The list of values of the objective function and the dual gap at each iteration Returned only if returncosts is True
niter int Number of iterations Returned only if returnniter is set to True

See also

GraphicalLasso GraphicalLassoCV

Notes

The algorithm employed to solve this problem is the GLasso algorithm from the Friedman 2008 Biostatistics paper It is the same algorithm as in the R glasso package
One possible difference with the glasso R package is that the diagonal coefficients are not penalized
66sklearn covariance Covariance Estimators 1561

scikitlearn user guide Release 0213

6611sklearncovariance ledoitwolf

sklearncovariance ledoitwolf XassumecenteredFalse blocksize1000

Estimates the shrunk LedoitWolf covariance matrix

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures Data from which to compute the covariance estimate

assumecentered boolean defaultFalse If True data will not be centered before computation Useful to work with data whose mean is significantly equal to zero but is not exactly zero If False data will be centered before computation

blocksize int default1000 Size of the blocks into which the covariance matrix will be split This is purely a memory optimization and does not affect results

Returns

shrunkcov arraylike shape nfeatures nfeatures Shrunk covariance

shrinkage float Coefficient in the convex combination used for the computation of the shrunk estimate

Notes

The regularized shrunk covariance is

$$\frac{1}{n} \text{shrinkage} \text{cov} + \frac{1 - \text{shrinkage}}{n} \text{np.identity}(n_{\text{features}})$$

where $\text{mu} = \text{trace}(\text{cov}) / n_{\text{features}}$

Examples using sklearncovarianceledoitwolf

•Sparse inverse covariance estimation

6612sklearncovariance oas

sklearncovariance oasXassumecenteredFalse

Estimate covariance with the Oracle Approximating Shrinkage algorithm

Parameters

Xarraylike shape nsamples nfeatures Data from which to compute the covariance estimate

assumecentered boolean If True data will not be centered before computation Useful to work with data whose mean is significantly equal to zero but is not exactly zero If False data will be centered before computation

Returns

shrunkcov arraylike shape nfeatures nfeatures Shrunk covariance

shrinkage float Coefficient in the convex combination used for the computation of the shrunk estimate

1562 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Notes

The regularised shrunk covariance is

$\frac{1}{n} \text{shrinkage} \cdot \text{cov} + \frac{1}{n} \text{shrinkage} \cdot \mu \cdot \text{npidentity} \cdot n \text{features}$

where $\mu = \frac{\text{trace}(\text{cov})}{n \text{features}}$

The formula we used to implement the OAS is slightly modified compared to the one given in the article See

OAS for more details

6613sklearncovariance shrunkcovariance

sklearncovariance shrunkcovariance empcov shrinkage01

Calculates a covariance matrix shrunk on the diagonal

Read more in the User Guide

Parameters

empcov arraylike shape nfeatures nfeatures Covariance matrix to be shrunk

shrinkage float 0 shrinkage 1 Coefficient in the convex combination used for the

computation of the shrunk estimate

Returns

shrunkcov arraylike Shrunk covariance

Notes

The regularized shrunk covariance is given by

$\frac{1}{n} \text{shrinkage} \cdot \text{cov} + \frac{1}{n} \text{shrinkage} \cdot \mu \cdot \text{npidentity} \cdot n \text{features}$

where $\mu = \frac{\text{trace}(\text{cov})}{n \text{features}}$

67sklearncrossdecomposition Cross decomposition

User guide See the Cross decomposition section for further details

crossdecompositionCCA ncomponents CCA Canonical Correlation Analysis

crossdecompositionPLSCanonical PLSCanonical implements the 2 blocks canonical PLS of

the original Wold algorithm Tenenhaus 1998 p204 re

ferred as PLSC2A in Wegelin 2000

crossdecompositionPLSRegression PLS regression

crossdecompositionPLSSVD ncomponents

Partial Least Square SVD

671sklearncrossdecomposition CCA

classsklearncrossdecomposition CCAncomponents2 scaleTrue maxiter500 tol1e06

copyTrue

CCA Canonical Correlation Analysis

67sklearncrossdecomposition Cross decomposition 1563

scikitlearn user guide Release 0213

CCA inherits from PLS with mode" B" and deflationmode"canonical"

Read more in the User Guide

Parameters

ncomponents int default 2 number of components to keep

scale boolean default True whether to scale the data

maxiter an integer default 500 the maximum number of iterations of the NIPALS inner loop

tolnonnegative real default 1e06 the tolerance used in the iterative algorithm

copy boolean Whether the deflation be done on a copy Let the default value to True unless you don't care about side effects

Attributes

xweights array p ncomponents X block weights vectors

yweights array q ncomponents Y block weights vectors

xloadings array p ncomponents X block loadings vectors

yloadings array q ncomponents Y block loadings vectors

xscores array nsamples ncomponents X scores

yscores array nsamples ncomponents Y scores

xrotations array p ncomponents X block to latents rotations

yrotations array q ncomponents Y block to latents rotations

niter arraylike Number of iterations of the NIPALS inner loop for each component

See also

PLSCanonical

PLSSVD

Notes

For each component k find the weights u v that maximizes $\max \text{corr}(X_k u, Y_k v)$ such that $u^T v = 1$

Note that it maximizes only the correlations between the scores

The residual matrix of X X_{k1} block is obtained by the deflation on the current X score xscore

The residual matrix of Y Y_{k1} block is obtained by deflation on the current Y score

References

Jacob A Wegelin A survey of Partial Least Squares PLS methods with emphasis on the twoblock case

Technical Report 371 Department of Statistics University of Washington Seattle 2000

In french but still a reference Tenenhaus M 1998 La regression PLS theorie et pratique Paris Editions Technic

1564 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

```
Examples
from sklearn.cross_decomposition import CCA
X = [[0, 0, 1, 100, 222, 354],
     [0, 1, 0, 2, 9, 11, 62, 59, 119, 123]]
Y = [[0, 1, 0, 2, 9, 11, 62, 59, 119, 123]]
cca = CCAncomponents1
ccafit = X, Y
```

```
CCACopy=True maxiter=500 ncomponents=1 scale=True tol=1e-06
Xc, Yc = cca.transform(X, Y)
```

```
Methods
fit(self, X, Y) Fit model to data
fit_transform(self, X, y) Learn and apply the dimension reduction on the train
data
get_params(self, deep) Get parameters for this estimator
predict(self, X, copy) Apply the dimension reduction learned on the train data
score(self, X, y, sample_weight) Returns the coefficient of determination R2 of the pre
diction
set_params(self, **params) Set the parameters of this estimator
transform(self, X, Y, copy) Apply the dimension reduction learned on the train data
init(self, ncomponents=2, scale=True, maxiter=500, tol=1e-06, copy=True)
fit(self, X, Y)
```

Fit model to data

```
Parameters
X: array-like shape (n_samples, n_features) Training vectors where n_samples is the
number of samples and n_features is the number of predictors
Y: array-like shape (n_samples, n_targets) Target vectors where n_samples is the num
ber of samples and n_targets is the number of response variables
fit_transform(self, X, y, None)
```

Learn and apply the dimension reduction on the train data

```
Parameters
X: array-like shape (n_samples, n_features) Training vectors where n_samples is the
number of samples and n_features is the number of predictors
Y: array-like shape (n_samples, n_targets) Target vectors where n_samples is the number
of samples and n_targets is the number of response variables
```

Returns

x_scores if Y is not given y_scores otherwise

```
get_params(self, deep=True)
Get parameters for this estimator
Parameters
```

deep: boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

scikitlearn user guide Release 0213

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXcopyTrue

Apply the dimension reduction learned on the train data

Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the

number of samples and nfeatures is the number of predictors

copy boolean default True Whether to copy X and Y or perform inplace normalization

Notes

This call requires the estimation of a p x q matrix which may be an issue in high dimensional space

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 0.23 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

1566 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

transform selfXYNone copyTrue

Apply the dimension reduction learned on the train data

Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of predictors

Yarraylike shape nsamples ntargets Target vectors where nsamples is the number of samples and ntargets is the number of response variables

copy boolean default True Whether to copy X and Y or perform inplace normalization

Returns

xscores if Y is not given xscores yscores otherwise

Examples using sklearncrossdecompositionCCA

- Multilabel classification

- Compare cross decomposition methods

672sklearncrossdecomposition PLSCanonical

classsklearncrossdecomposition PLSCanonical ncomponents2 scaleTrue algorithm'nipals' maxiter500 tol1e06

copyTrue
PLSCanonical implements the 2 blocks canonical PLS of the original Wold algorithm Tenenhaus 1998 p204 referred as PLSC2A in Wegelin 2000

This class inherits from PLS with mode"A" and deflationmode"canonical" normyweightsTrue and algorithm"nipals" but svd should provide similar results up to numerical errors

Read more in the User Guide

Parameters

ncomponents int default 2 Number of components to keep

scale boolean default True Option to scale data

algorithm string "nipals" or "svd" The algorithm used to estimate the weights It will be called ncomponents times ie once for each iteration of the outer loop

maxiter an integer default 500 the maximum number of iterations of the NIPALS inner loop used only if algorithm"nipals"

tolnonnegative real default 1e06 the tolerance used in the iterative algorithm

copy boolean default True Whether the deflation should be done on a copy Let the default value to True unless you don't care about side effect

Attributes

xweights array shape p ncomponents X block weights vectors

yweights array shape q ncomponents Y block weights vectors

xloadings array shape p ncomponents X block loadings vectors

yloadings array shape q ncomponents Y block loadings vectors

xscores array shape nsamples ncomponents X scores

67sklearncrossdecomposition Cross decomposition 1567

scikitlearn user guide Release 0213

yscores array shape nsamples ncomponents Y scores

xrotations array shape p ncomponents X block to latents rotations

yrotations array shape q ncomponents Y block to latents rotations

niter arraylike Number of iterations of the NIPALS inner loop for each component Not useful if the algorithm provided is “svd”

See also

CCA

PLSSVD

Notes

Matrices

T xscores

U yscores

W xweights

C yweights

P xloadings

Q yloadings

Are computed such that

$X^T P T^T \text{Err}$ and $Y^T U Q T^T \text{Err}$

$T_k^T X_k W_k$ for k in $\text{rang}(\text{ncomponents})$

$U_k^T Y_k C_k$ for k in $\text{rang}(\text{ncomponents})$

xrotations $W P T W^T$

yrotations $C Q T C^T$

where X_k and Y_k are residual matrices at iteration k

Slides explaining PLS

For each component k find weights u, v that optimize

$\max \text{corr}(X_k u, Y_k v) / \sqrt{\text{std}(X_k u)^2 + \text{std}(Y_k v)^2}$ such that $u^T v = 1$

Note that it maximizes both the correlations between the scores and the intrablock variances

The residual matrix of X X_{k+1} block is obtained by the deflation on the current X score x_{score}

The residual matrix of Y Y_{k+1} block is obtained by deflation on the current Y score This performs a canonical symmetric version of the PLS regression But slightly different than the CCA This is mostly used for modeling

This implementation provides the same results that the “plsrm” package provided in the R language R

project using the function `plsca(X, Y)` Results are equal or collinear with the function `pls mode`

canonical of the “mixOmics” package The difference relies in the fact that mixOmics implementation does not exactly implement the Wold algorithm since it does not normalize y_{weights} to one

References

Jacob A Wegelin A survey of Partial Least Squares PLS methods with emphasis on the twoblock case

Technical Report 371 Department of Statistics University of Washington Seattle 2000

1568 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
Tenenhaus M 1998 La regression PLS theorie et pratique Paris Editions Technic
Examples

```
from sklearn.crossdecomposition import PLSCanonical
X = [[0, 0, 1, 100, 222, 254],
     [0, 1, 0, 2, 0, 9, 11, 62, 59, 119, 123]]
Y = [0, 1, 0, 2, 0, 9, 11, 62, 59, 119, 123]
plsca = PLSCanonical(ncomponents=2)
plsca.fit(X, Y)

PLSCanonical.algorithm = 'nipals'
plsca.copy = True
plsca.max_iter = 500
plsca.ncomponents = 2
plsca.scale = True
plsca.tol = 1e-06

Xc, Yc = plsca.transform(X, Y)

Methods
plsca.fit(X, Y) Fit model to data
plsca.fit_transform(X, y) Learn and apply the dimension reduction on the train data
plsca.get_params() Get parameters for this estimator
plsca.predict(X) Apply the dimension reduction learned on the train data
plsca.score(X, y, sample_weight) Returns the coefficient of determination R2 of the prediction
plsca.set_params(**params) Set the parameters of this estimator
plsca.transform(X) Apply the dimension reduction learned on the train data
plsca.init(ncomponents=2, scale=True, algorithm='nipals', max_iter=500, tol=1e-06, copy=True)
plsca.fit(X, Y) Fit model to data

Parameters
X: array-like shape (n_samples, n_features) Training vectors where n_samples is the number of samples and n_features is the number of predictors
Y: array-like shape (n_samples, n_targets) Target vectors where n_samples is the number of samples and n_targets is the number of response variables
plsca.fit_transform(X, y) Learn and apply the dimension reduction on the train data

Parameters
X: array-like shape (n_samples, n_features) Training vectors where n_samples is the number of samples and n_features is the number of predictors
y: array-like shape (n_samples, n_targets) Target vectors where n_samples is the number of samples and n_targets is the number of response variables
Returns
x_scores: if Y is not given x_scores y_scores otherwise
67sklearn.crossdecomposition Cross decomposition 1569
```

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXcopyTrue

Apply the dimension reduction learned on the train data

Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of predictors

copy boolean default True Whether to copy X and Y or perform inplace normalization

Notes

This call requires the estimation of a p x q matrix which may be an issue in high dimensional space

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

1570 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfXYNone copyTrue

Apply the dimension reduction learned on the train data

Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of predictors

Yarraylike shape nsamples ntargets Target vectors where nsamples is the number of samples and ntargets is the number of response variables

copy boolean default True Whether to copy X and Y or perform inplace normalization

Returns

xscores if Y is not given xscores yscores otherwise

Examples using sklearncrossdecompositionPLSCanonical

•Compare cross decomposition methods

673sklearncrossdecomposition PLSRegression

classsklearncrossdecomposition PLSRegression ncomponents2 scaleTrue

maxiter500 tol1e06 copyTrue

PLS regression

PLSRegression implements the PLS 2 blocks regression known as PLS2 or PLS1 in case of one dimensional response This class inherits from PLS with mode"A" deflationmode"regression" normyweightsFalse and algorithm"nipals"

Read more in the User Guide

Parameters

ncomponents int default 2 Number of components to keep

scale boolean default True whether to scale the data

maxiter an integer default 500 the maximum number of iterations of the NIPALS inner loop used only if algorithm"nipals"

tolnonnegative real Tolerance used in the iterative algorithm default 1e06

copy boolean default True Whether the deflation should be done on a copy Let the default value to True unless you don't care about side effect

Attributes

xweights array p ncomponents X block weights vectors

yweights array q ncomponents Y block weights vectors

xloadings array p ncomponents X block loadings vectors

yloadings array q ncomponents Y block loadings vectors

67sklearncrossdecomposition Cross decomposition 1571

scikitlearn user guide Release 0213

xscores array nsamples ncomponents X scores

yscores array nsamples ncomponents Y scores

xrotations array p ncomponents X block to latents rotations

yrotations array q ncomponents Y block to latents rotations

coef array p q The coefficients of the linear model $Y = X \text{coef} + \text{Err}$

niter arraylike Number of iterations of the NIPALS inner loop for each component

Notes

Matrices

T xscores

U yscores

W xweights

C yweights

P xloadings

Q yloadings

Are computed such that

$X = T P^T$ and $Y = U Q^T$

$T_k = X_k W_k$ for k in $range(ncomponents)$

$U_k = Y_k C_k$ for k in $range(ncomponents)$

$xrotations = W P^T W^T$

$yrotations = C Q^T C^T$

where X_k and Y_k are residual matrices at iteration k

Slides explaining PLS

For each component k find weights u, v that optimizes $\max \text{corr}(X_k u, Y_k v)$ s.t. $\|u\| = 1$

Note that it maximizes both the correlations between the scores and the intrablock variances

The residual matrix of X X_{k+1} block is obtained by the deflation on the current X score x_{score}

The residual matrix of Y Y_{k+1} block is obtained by deflation on the current X score This performs the PLS

regression known as PLS2 This mode is prediction oriented

This implementation provides the same results that 3 PLS packages provided in the R language Rproject

- “mixOmics” with function `plsX` Y mode “regression”

- “pls” with function `plsreg2` X, Y

- “pls” with function `oscoresplsfit` X, Y

References

Jacob A Wegelin A survey of Partial Least Squares PLS methods with emphasis on the twoblock case

Technical Report 371 Department of Statistics University of Washington Seattle 2000

In french but still a reference Tenenhaus M 1998 La regression PLS theorie et pratique Paris Editions

Technic

1572 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

from sklearn.cross_decomposition import PLSRegression

X = [[0, 0, 1, 100, 222, 254]

Y = [[0, 1, 0, 2, 0, 9, 11, 62, 59, 119, 123]

pls2 = PLSRegression(n_components=2

pls2.fit(X, Y)

PLSRegression(copy=True, max_iter=500, n_components=2, scale=True,

tol=1e-06,

Y_pred = pls2.predict(X)

Methods

fit(self, X, Y) Fit model to data

fit_transform(self, X, y) Learn and apply the dimension reduction on the train

data

get_params(self, deep=True) Get parameters for this estimator

predict(self, X, copy=True) Apply the dimension reduction learned on the train data

score(self, X, y, sample_weight) Returns the coefficient of determination R² of the pre-

diction

set_params(self, **kwargs) Set the parameters of this estimator

transform(self, X, Y, copy=True) Apply the dimension reduction learned on the train data

init(self, n_components=2, scale=True, max_iter=500, tol=1e-06, copy=True,

fit_self_XY=True)

Fit model to data

Parameters

X: array-like shape (n_samples, n_features) Training vectors where n_samples is the

number of samples and n_features is the number of predictors

Y: array-like shape (n_samples, n_targets) Target vectors where n_samples is the num-

ber of samples and n_targets is the number of response variables

fit_transform(self, X, y, None)

Learn and apply the dimension reduction on the train data

Parameters

X: array-like shape (n_samples, n_features) Training vectors where n_samples is the

number of samples and n_features is the number of predictors

Y: array-like shape (n_samples, n_targets) Target vectors where n_samples is the number

of samples and n_targets is the number of response variables

Returns

X_scores: if Y is not given X_scores, Y_scores otherwise

get_params(self, deep=True)

Get parameters for this estimator

Parameters

67sklearn.cross_decomposition Cross decomposition 1573

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXcopyTrue

Apply the dimension reduction learned on the train data

Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of predictors

copy boolean default True Whether to copy X and Y or perform inplace normalization

Notes

This call requires the estimation of a $p \times q$ matrix which may be an issue in high dimensional space

scoreselfXysampleweightNone

Returns the coefficient of determination R^2 of the prediction

The coefficient R^2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R^2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R^2 of selfpredictX wrt y

Notes

The R^2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

1574 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

self
transform selfXYNone copyTrue
Apply the dimension reduction learned on the train data
Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of predictors
Yarraylike shape nsamples ntargets Target vectors where nsamples is the number of samples and ntargets is the number of response variables
copy boolean default True Whether to copy X and Y or perform inplace normalization

Returns
xscores if Y is not given xscores yscores otherwise
Examples using sklearncrossdecompositionPLSRegression
•Compare cross decomposition methods
674sklearncrossdecomposition PLSSVD
classsklearncrossdecomposition PLSSVDncomponents2 scaleTrue copyTrue
Partial Least Square SVD
Simply perform a svd on the crosscovariance matrix X'Y There are no iterative deflation here
Read more in the User Guide

Parameters
ncomponents int default 2 Number of components to keep
scale boolean default True Whether to scale X and Y
copy boolean default True Whether to copy X and Y or perform inplace computations
Attributes
xweights array p ncomponents X block weights vectors
yweights array q ncomponents Y block weights vectors
xscores array nsamples ncomponents X scores
yscores array nsamples ncomponents Y scores

See also
PLSCanonical
CCA
Examples
67sklearncrossdecomposition Cross decomposition 1575

```
scikitlearn user guide Release 0213
import numpy as np
from sklearn.cross_decomposition import PLSSVD
X = np.array([[0, 1],
              [100,
               222,
               254],
              [0, 1],
              [0, 1],
              [62, 59],
              [119, 123]])
Y = np.array([[0, 1],
              [0, 1],
              [62, 59],
              [119, 123]])
plsca = PLSSVD(n_components=2)
plsca.fit(X, Y)
PLSSVD(copy=True, n_components=2, scale=True)
Xc, Yc = plsca.transform(X, Y)
Xc.shape, Yc.shape
(4, 2), (4, 2)
Methods
fitself(X, Y) Fit model to data
fit_transform(self, X, y) Learn and apply the dimension reduction on the train
data
get_params(self, deep=True) Get parameters for this estimator
set_params(self, **params) Set the parameters of this estimator
transform(self, X, Y) Apply the dimension reduction learned on the train data
init(self, n_components=2, scale=True, copy=True)
fitself(X, Y)
Fit model to data
Parameters
X : array-like, shape (n_samples, n_features) Training vectors, where n_samples is the
number of samples and n_features is the number of predictors
Y : array-like, shape (n_samples, n_targets) Target vectors, where n_samples is the num
ber of samples and n_targets is the number of response variables
fit_transform(self, X, y)
Learn and apply the dimension reduction on the train data
Parameters
X : array-like, shape (n_samples, n_features) Training vectors, where n_samples is the
number of samples and n_features is the number of predictors
y : array-like, shape (n_samples, n_targets) Target vectors, where n_samples is the number
of samples and n_targets is the number of response variables
Returns
Xscores if Y is not given, xscores, yscores otherwise
get_params(self, deep=True)
Get parameters for this estimator
Parameters
1576 Chapter 6 API (Application Programming Interface) Reference
```

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfXYNone

Apply the dimension reduction learned on the train data

Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of predictors

Yarraylike shape nsamples ntargets Target vectors where nsamples is the number of samples and ntargets is the number of response variables

68sklearndatasets Datasets

The sklearndatasets module includes utilities to load datasets including methods to load and fetch popular reference datasets It also features some artificial data generators

User guide See the Dataset loading utilities section for further details

681 Loaders

datasetscleardatahome datahome Delete all the content of the data home cache

datasetsdumpsVMLightfile X y f Dump the dataset in svmlight libsvm file format

datasetsfetch20newsgroups datahome

Load the filenames and data from the 20 newsgroups

dataset classification

datasetsfetch20newsgroupsvectorized Load the 20 newsgroups dataset and vectorize it into token counts classification

datasetsfetchcaliforniahousing Load the California housing dataset regression

datasetsfetchcovtype datahome Load the covtype dataset classification

datasetsfetchkddcup99 subset datahome

Load the kddcup99 dataset classification

datasetsfetchlfwpairs subset Load the Labeled Faces in the Wild LFW pairs dataset classification

datasetsfetchlfwpeople datahome Load the Labeled Faces in the Wild LFW people dataset classification

datasetsfetcholivettifaces datahome

Load the Olivetti faces dataset from ATT classification

datasetsfetchopenml name version Fetch dataset from openml by name or dataset id

Continued on next page

68sklearndatasets Datasets 1577

scikitlearn user guide Release 0213

Table 647 – continued from previous page

`datasetsfetchrcv1` `datahome` `subset` Load the RCV1 multilabel dataset classification

`datasetsfetchspeciesdistributions` Loader for species distribution dataset from Phillips et

`datasetsgetdatahome` `datahome` Return the path of the scikitlearn data dir

`datasetsloadboston` `returnXy` Load and return the boston houseprices dataset regres
sion

`datasetsloadbreastcancer` `returnXy` Load and return the breast cancer wisconsin dataset clas
sification

`datasetsloaddiabetes` `returnXy` Load and return the diabetes dataset regression

`datasetsloaddigits` `nclass` `returnXy` Load and return the digits dataset classification

`datasetsloadfiles` `containerpath` Load text files with categories as subfolder names

`datasetsloadiris` `returnXy` Load and return the iris dataset classification

`datasetsloadlinnerud` `returnXy` Load and return the linnerud dataset multivariate regres
sion

`datasetsloadsampleimage` `imagename` Load the numpy array of a single sample image

`datasetsloadsampleimages` Load sample images for image manipulation

`datasetsloadsvmlightfile` `f` `nfeatures`
Load datasets in the svmlight libsvm format into sparse
CSR (Compressed Sparse Row) matrix

`datasetsloadsvmlightfiles` `files` Load dataset from multiple files in SVMlight format

`datasetsloadwine` `returnXy` Load and return the wine dataset classification

`sklearn.datasets.cleardatahome`
`sklearn.datasets.cleardatahome` `datahome` `None`
Delete all the content of the data home cache

Parameters

`datahome` `str` `None` The path to scikitlearn data dir

`sklearn.datasets.dumpsVMLightfile`
`sklearn.datasets.dumpsVMLightfile` `Xy` `f` `zerobased` `True` `comment` `None`
`queryid` `None` `multilabel` `False`
Dump the dataset in svmlight libsvm file format

This format is a textbased format with one sample per line It does not store zero valued features hence is
suitable for sparse dataset

The first element of each line can be used to store a target variable to predict

Parameters

`X` `arraylike` `sparse matrix` `shape` `nsamples` `nfeatures` Training vectors where
`nsamples` is the number of samples and `nfeatures` is the number of features

`y` `arraylike` `sparse matrix` `shape` `nsamples` `nlabels` Target values Class labels
must be an integer or float or arraylike objects of integer or float for multilabel classifica
tions

`fstring` or `filelike` in binary mode If string specifies the path that will contain the data If
`filelike` data will be written to `f` `f` should be opened in binary mode

`zerobased` `boolean` optional Whether column indices should be written `zerobased` `True` or
`onebased` `False`

`comment` `string` optional Comment to insert at the top of the file This should be either a
Unicode string which will be encoded as UTF8 or an ASCII byte string If a comment

1578 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0.21.3

is given then it will be preceded by one that identifies the file as having been dumped by scikitlearn. Note that not all tools grok comments in SVMlight files.

queryid arraylike shape n_samples Array containing pairwise preference constraints

qid in svmformat

multilabel boolean optional Samples may have several labels each see <https://www.cs.cmu.edu/~dwj/libsvmtools/datasets/multilabel.html>

New in version 0.17 parameter multilabel to support multilabel datasets

Examples using sklearn.datasets.dump_svmlight_file

- Libsvm GUI

sklearn.datasets.fetch_20newsgroups

sklearn.datasets.fetch_20newsgroups(data_home=None, subset='train', categories=None, shuffle=True, random_state=42, remove_duplicate_headers=True, load_if_missing=True)

Load the filenames and data from the 20 newsgroups dataset classification

Download it if necessary

Classes 20

Samples total 18846

Dimensionality 1

Features text

Read more in the User Guide

Parameters

data_home optional default None Specify a download and cache folder for the datasets. If None all scikitlearn data is stored in 'scikitlearn_data' subfolders

subset 'train' or 'test' 'all' optional Select the dataset to load 'train' for the training set 'test' for the test set 'all' for both with shuffled ordering

categories None or collection of string or unicode If None default load all the categories. If not None list of category names to load other categories ignored

shuffle bool optional Whether or not to shuffle the data might be important for models that make the assumption that the samples are independent and identically distributed iid such as stochastic gradient descent

random_state int RandomState instance or None default Determines random number generation for dataset shuffling. Pass an int for reproducible output across multiple function calls. See Glossary

remove_duplicate_headers tuple May contain any subset of 'headers' 'footers' 'quotes'. Each of these are kinds of text that will be detected and removed from the newsgroup posts preventing classifiers from overfitting on metadata

'headers' removes newsgroup headers 'footers' removes blocks at the ends of posts that look like signatures and 'quotes' removes lines that appear to be quoting another post

'headers' follows an exact standard the other filters are not always correct

68 sklearn.datasets Datasets 1579

scikitlearn user guide Release 0213
downloadifmissing optional True by default If False raise an IOError if the data is not locally available instead of trying to download the data from the source site
Returns

bunch Bunch object with the following attribute

- bunchdata list length nsamples
- bunchtarget array shape nsamples
- bunchfilenames list length nsamples
- bunchDESCR a description of the dataset
- bunchtargetnames a list of categories of the returned data length nclasses This depends on the categories parameter

Examples using sklearn.datasets.fetch20newsgroups

- Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation
- Biclustering documents with the Spectral Coclustering algorithm
- Column Transformer with Heterogeneous Data Sources
- Sample pipeline for text feature extraction and evaluation
- FeatureHasher and DictVectorizer Comparison
- Clustering text documents using kmeans
- Classification of text documents using sparse features

sklearn.datasets.fetch20newsgroupsvectorized
sklearn.datasets.fetch20newsgroupsvectorized subset='train' remove
datahome=None down
loadifmissing=True re
turnXy=False

Load the 20 newsgroups dataset and vectorize it into token counts classification
Download it if necessary
This is a convenience function the transformation is done using the default settings for sklearn
feature_extraction.text.CountVectorizer For more advanced usage stopword
filtering ngram extraction etc combine fetch20newsgroups with a custom sklearn
feature_extraction.text.CountVectorizer sklearn.feature_extraction.text
HashingVectorizer sklearn.feature_extraction.text.TfidfTransformer or
sklearn.feature_extraction.text.TfidfVectorizer

Classes 20
Samples total 18846
Dimensionality 130107
Features real
Read more in the User Guide
Parameters
1580 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

subset 'train' or 'test' 'all' optional Select the dataset to load 'train' for the training set 'test' for the test set 'all' for both with shuffled ordering
remove tuple May contain any subset of 'headers' 'footers' 'quotes' Each of these are kinds of text that will be detected and removed from the newsgroup posts preventing classifiers from overfitting on metadata
'headers' removes newsgroup headers 'footers' removes blocks at the ends of posts that look like signatures and 'quotes' removes lines that appear to be quoting another post
datahome optional default None Specify a download and cache folder for the datasets If None all scikitlearn data is stored in 'scikitlearndata' subfolders
downloadifmissing optional True by default If False raise an IOError if the data is not locally available instead of trying to download the data from the source site
returnXy boolean default False If True returns datadata datatarget in stead of a Bunch object

New in version 020

Returns

bunch Bunch object with the following attribute

- bunchdata sparse matrix shape nsamples nfeatures
- bunchtarget array shape nsamples
- bunchtargetnames a list of categories of the returned data length nclasses
- bunchDESCR a description of the dataset

data target tuple ifreturnXy is True New in version 020

Examples using sklearndatasetsfetch20newsgroupsvectorized

- The JohnsonLindenstrauss bound for embedding with random projections
- Model Complexity Influence
- Multiclass sparse logisitic regression on newgroups20

sklearndatasets fetchcaliforniahousing

sklearndatasets fetchcaliforniahousing datahomeNone downloadifmissingTrue
returnXyFalse

Load the California housing dataset regression

Samples total 20640

Dimensionality 8

Features real

Target real 015 5

Read more in the User Guide

Parameters

datahome optional default None Specify another download and cache folder for the datasets By default all scikitlearn data is stored in 'scikitlearndata' subfolders
68sklearndatasets Datasets 1581

scikitlearn user guide Release 0213

downloadifmissing optional defaultTrue If False raise a IOError if the data is not locally available instead of trying to download the data from the source site

returnXy boolean defaultFalse If True returns datadata datatarget in stead of a Bunch object

New in version 020

Returns

dataset dictlike object with the following attributes

datasetdata ndarray shape 20640 8 Each row corresponding to the 8 feature values in order

datasettarget numpy array of shape 20640 Each value corresponds to the average house value in units of 100000

datasetfeaturenames array of length 8 Array of ordered feature names used in the dataset

datasetDESCR string Description of the California housing dataset

data target tuple ifreturnXy is True New in version 020

Notes

This dataset consists of 20640 samples and 9 features

Examples using sklearndatasetsfetchcaliforniahousing

- Imputing missing values with variants of IterativeImputer
- Partial Dependence Plots
- Compare the effect of different scalers on data with outliers

sklearndatasets fetchcovtype

sklearndatasets fetchcovtype datahomeNone downloadifmissingTrue ran

domstateNone shuffleFalse returnXyFalse

Load the covtype dataset classification

Download it if necessary

Classes 7

Samples total 581012

Dimensionality 54

Features int

Read more in the User Guide

Parameters

datahome string optional Specify another download and cache folder for the datasets By default all scikitlearn data is stored in ‘scikitlearndata’ subfolders

downloadifmissing boolean defaultTrue If False raise a IOError if the data is not locally available instead of trying to download the data from the source site

1582 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

randomstate int RandomState instance or None default Determines random number generation for dataset shuffling Pass an int for reproducible output across multiple function calls See Glossary

shuffle bool defaultFalse Whether to shuffle dataset

returnXy boolean defaultFalse If True returns datadata datatarget in stead of a Bunch object

New in version 020

Returns

dataset dictlike object with the following attributes

datasetdata numpy array of shape 581012 54 Each row corresponds to the 54 features in the dataset

datasettarget numpy array of shape 581012 Each value corresponds to one of the 7 forest covertypes with values ranging between 1 to 7

datasetDESCR string Description of the forest covertype dataset

data target tuple ifreturnXy is True New in version 020

sklearndatasets fetchkddcup99

sklearndatasets fetchkddcup99 subsetNone datahomeNone shuffleFalse

randomstateNone percent10True down

loadifmissingTrue returnXyFalse

Load the kddcup99 dataset classification

Download it if necessary

Classes 23

Samples total 4898431

Dimensionality 41

Features discrete int or continuous float

Read more in the User Guide

New in version 018

Parameters

subset None 'SA' 'SF' 'http' 'smtp' To return the corresponding classical subsets of kddcup 99 If None return the entire kddcup 99 dataset

datahome string optional Specify another download and cache folder for the datasets By default all scikitlearn data is stored in 'scikitlearndata' subfolders versionadded 019

shuffle bool defaultFalse Whether to shuffle dataset

randomstate int RandomState instance or None default Determines random number generation for dataset shuffling and for selection of abnormal samples if subsetSA Pass an int for reproducible output across multiple function calls See Glossary

percent10 bool defaultTrue Whether to load only 10 percent of the data

downloadifmissing bool defaultTrue If False raise a IOError if the data is not locally available instead of trying to download the data from the source site

68sklearndatasets Datasets 1583

scikitlearn user guide Release 0213

returnXy boolean defaultFalse If True returns data target instead of a Bunch object See below for more information about the data andtarget object

New in version 020

Returns

data Bunch

Dictionarylike object the interesting attributes are

- ‘data’ the data to learn
- ‘target’ the regression target for each sample
- ‘DESCR’ a description of the dataset

data target tuple ifreturnXy is True New in version 020

sklearn.datasets.fetchlfwpairs

sklearn.datasets.fetchlfwpairs subset‘train’ datahomeNone funneledTrue re

size05 colorFalse sliceslice70 195None slice78

172None downloadifmissingTrue

Load the Labeled Faces in the Wild LFW pairs dataset classification

Download it if necessary

Classes 5749

Samples total 13233

Dimensionality 5828

Features real between 0 and 255

In the official README.txt this task is described as the “Restricted” task As I am not sure as to implement the “Unrestricted” variant correctly I left it as unsupported for now

The original images are 250 x 250 pixels but the default slice and resize arguments reduce them to 62 x 47

Read more in the User Guide

Parameters

subset optional default ‘train’ Select the dataset to load ‘train’ for the development training

set ‘test’ for the development test set and ‘10folds’ for the official evaluation set that is

meant to be used with a 10folds cross validation

datahome optional default None Specify another download and cache folder for the

datasets By default all scikitlearn data is stored in ‘scikitlearn_data’ subfolders

funneled boolean optional default True Download and use the funneled variant of the

dataset

resize float optional default 0.5 Ratio used to resize the each face picture

color boolean optional default False Keep the 3 RGB channels instead of averaging them to a single gray level channel If color is True the shape of the data has one more dimension than the shape with color False

slice optional Provide a custom 2D slice height width to extract the ‘interesting’ part of

the jpeg files and avoid use statistical correlation from the background

1584 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

downloadifmissing optional True by default If False raise a IOError if the data is not locally available instead of trying to download the data from the source site

Returns

The data is returned as a Bunch object with the following attributes

data numpy array of shape 2200 5828 Shape depends on subset Each row corresponds to 2 ravel'd face images of original size 62 x 47 pixels Changing the slice resize or subset parameters will change the shape of the output

pairs numpy array of shape 2200 2 62 47 Shape depends on subset Each row has 2 face images corresponding to same or different person from the dataset containing 5749 people Changing the slice resize or subset parameters will change the shape of the output

target numpy array of shape 2200 Shape depends on subset Labels associated to each pair of images The two label values being different persons or the same person

DESCR string Description of the Labeled Faces in the Wild LFW dataset

sklearn.datasets.fetchlfwpeople

sklearn.datasets.fetchlfwpeople datahomeNone funneledTrue resize05

minfacesperperson0 colorFalse sliceslice70

195 None slice78 172 None down

loadifmissingTrue returnXyFalse

Load the Labeled Faces in the Wild LFW people dataset classification

Download it if necessary

Classes 5749

Samples total 13233

Dimensionality 5828

Features real between 0 and 255

Read more in the User Guide

Parameters

datahome optional default None Specify another download and cache folder for the datasets By default all scikitlearn data is stored in 'scikitlearn\data' subfolders

funneled boolean optional default True Download and use the funneled variant of the dataset

resize float optional default 0.5 Ratio used to resize the each face picture

minfacesperperson int optional default None The extracted dataset will only retain pictures of people that have at least minfacesperperson different pictures

color boolean optional default False Keep the 3 RGB channels instead of averaging them to a single gray level channel If color is True the shape of the data has one more dimension than the shape with color False

slice optional Provide a custom 2D slice height width to extract the 'interesting' part of the jpeg files and avoid use statistical correlation from the background

downloadifmissing optional True by default If False raise a IOError if the data is not locally available instead of trying to download the data from the source site

68sklearn.datasets Datasets 1585

scikitlearn user guide Release 0213

returnXy boolean defaultFalse If True returns datasetdata dataset

target instead of a Bunch object See below for more information about the dataset

data anddatasettarget object

New in version 020

Returns

dataset dictlike object with the following attributes

datasetdata numpy array of shape 13233 2914 Each row corresponds to a ravelled face

image of original size 62 x 47 pixels Changing the slice or resize parameters will

change the shape of the output

datasetimages numpy array of shape 13233 62 47 Each row is a face image corresponding

to one of the 5749 people in the dataset Changing the slice or resize parameters will

change the shape of the output

datasettarget numpy array of shape 13233 Labels associated to each face image Those

labels range from 05748 and correspond to the person IDs

datasetDESCR string Description of the Labeled Faces in the Wild LFW dataset

data target tuple ifreturnXy is True New in version 020

Examples using sklearndatasetsfetchlfwpeople

•Faces recognition example using eigenfaces and SVMs

sklearndatasets fetchholivettifaces

sklearndatasets fetchholivettifaces datahomeNone shuffleFalse randomstate0

downloadifmissingTrue

Load the Olivetti faces dataset from ATT classification

Download it if necessary

Classes 40

Samples total 400

Dimensionality 4096

Features real between 0 and 1

Read more in the User Guide

Parameters

datahome optional default None Specify another download and cache folder for the

datasets By default all scikitlearn data is stored in ‘scikitlearndata’ subfolders

shuffle boolean optional If True the order of the dataset is shuffled to avoid having images of

the same person grouped

randomstate int RandomState instance or None default0 Determines random number

generation for dataset shuffling Pass an int for reproducible output across multiple function

calls See Glossary

downloadifmissing optional True by default If False raise a IOError if the data is not

locally available instead of trying to download the data from the source site

1586 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

An object with the following attributes

data numpy array of shape 400 4096 Each row corresponds to a ravelled face image of original size 64 x 64 pixels

images numpy array of shape 400 64 64 Each row is a face image corresponding to one of the 40 subjects of the dataset

target numpy array of shape 400 Labels associated to each face image Those labels are ranging from 039 and correspond to the Subject IDs

DESCR string Description of the modified Olivetti Faces Dataset

Examples using sklearn.datasets.fetcholivettifaces

- Face completion with a multioutput estimators
- Online learning of a dictionary of parts of faces
- Faces dataset decompositions
- Pixel importances with a parallel forest of trees

sklearn.datasets.fetchopenml

sklearn.datasets.fetchopenml nameNone version'active' dataidNone datahomeNone

targetcolumn'default'target' cacheTrue returnXyFalse

Fetch dataset from openml by name or dataset id

Datasets are uniquely identified by either an integer ID or by a combination of name and version ie there might be multiple versions of the 'iris' dataset Please give either name or dataid not both In case a name is given a version can also be provided

Read more in the User Guide

Note EXPERIMENTAL

The API (Application Programming Interface) is experimental particularly the return value structure and might have small backwardincompatible changes in future releases

Parameters

name str or None String identifier of the dataset Note that OpenML can have multiple datasets with the same name

version integer or 'active' default'active' Version of the dataset Can only be provided if alsoname is given If 'active' the oldest version that's still active is used Since there may be more than one active version of a dataset and those versions may fundamentally be different from one another setting an exact version is highly recommended

dataid int or None OpenML ID of the dataset The most specific way of retrieving a dataset

If dataid is not given name and potential version are used to obtain a dataset

datahome string or None default None Specify another download and cache folder for the data sets By default all scikitlearn data is stored in 'scikitlearn\data' subfolders

68sklearn.datasets Datasets 1587

scikitlearn user guide Release 0213

targetcolumn string list or None default 'defaulttarget' Specify the column name in the data to use as target If 'defaulttarget' the standard target column a stored on the server is used IfNone all columns are returned as data and the target is None If list of strings all columns with these names are returned as multitarget Note not all scikitlearn classifiers can handle all types of multioutput combinations
cache boolean defaultTrue Whether to cache downloaded datasets using joblib
returnXy boolean defaultFalse If True returns data target instead of a Bunch object See below for more information about the data andtarget objects

Returns
data Bunch Dictionarylike object with attributes
data nparray or scipysparsematrix of floats The feature matrix Categorical features are encoded as ordinals
target nparray The regression target or classification labels if applicable Dtype is float if numeric and object if categorical
DESCR str The full description of the dataset
featurenames list The names of the dataset columns
categories dict Maps each categorical feature name to a list of values such that the value encoded as i is ith in the list
details dict More metadata from OpenML
data target tuple ifreturnXy is True

Note EXPERIMENTAL
This interface is experimental and subsequent releases may change attributes without notice although there should only be minor changes to data andtarget
Missing values in the 'data' are represented as NaN's Missing values in 'target' are represented as NaN's numerical target or None categorical target

Examples using sklearndatasetsfetchopenml
•Gaussian process regression GPR on Mauna Loa CO2 data
•MNIST classification using multinomial logistic L1
•Early stopping of Stochastic Gradient Descent
•Classifier Chain
•Visualization of MLP weights on MNIST

sklearndatasets fetchrcv1
sklearndatasets fetchrcv1 datahomeNone subset'all' downloadifmissingTrue ran
domstateNone shuffleFalse returnXyFalse
Load the RCV1 multilabel dataset classification
Download it if necessary

1588 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
Version RCV1v2 vectors full sets topics multilabels
Classes 103
Samples total 804414
Dimensionality 47236
Features real between 0 and 1
Read more in the User Guide
New in version 017
Parameters
datahome string optional Specify another download and cache folder for the datasets By default all scikitlearn data is stored in 'scikitlearn\data' subfolders
subset string 'train' 'test' or 'all' default'all' Select the dataset to load 'train' for the training set 23149 samples 'test' for the test set 781265 samples 'all' for both with the training samples first if shuffle is False This follows the official LYRL2004 chronological split
downloadifmissing boolean defaultTrue If False raise a IOError if the data is not locally available instead of trying to download the data from the source site
randomstate int RandomState instance or None default Determines random number generation for dataset shuffling Pass an int for reproducible output across multiple function calls See Glossary
shuffle bool defaultFalse Whether to shuffle dataset
returnXy boolean defaultFalse If True returns dataset\data dataset
target instead of a Bunch object See below for more information about the dataset
data anddataset\target object
New in version 020
Returns
dataset dictlike object with the following attributes
dataset\data scipy csr array dtype npfloat64 shape 804414 47236 The array has 016 of non zero values
dataset\target scipy csr array dtype npuint8 shape 804414 103 Each sample has a value of 1 in its categories and 0 in others The array has 315 of non zero values
dataset\sampleid numpy array dtype npuint32 shape 804414 Identification number of each sample as ordered in dataset\data
dataset\targetnames numpy array dtype object length 103 Names of each target RCV1 topics as ordered in dataset\target
dataset\DESCR string Description of the RCV1 dataset
data target tuple ifreturnXy is True New in version 020
sklearn\datasets fetchspeciesdistributions
sklearn\datasets fetchspeciesdistributions datahomeNone down
loadifmissingTrue
Loader for species distribution dataset from Phillips et al 2006
68sklearn\datasets Datasets 1589

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

datahome optional default None Specify another download and cache folder for the datasets By default all scikitlearn data is stored in 'scikitlearndata' subfolders

downloadifmissing optional True by default If False raise a IOError if the data is not locally available instead of trying to download the data from the source site

Returns

The data is returned as a Bunch object with the following attributes

coverages array shape 14 1592 1212 These represent the 14 features measured at each point of the map grid The latitudelongitude values for the grid are discussed below Missing data is represented by the value 9999

train record array shape 1624 The training points for the data Each point has three fields

- train'species' is the species name
- train'dd long' is the longitude in degrees
- train'dd lat' is the latitude in degrees

test record array shape 620 The test points for the data Same format as the training data

Nx Ny integers The number of longitudes x and latitudes y in the grid

xleftlowercorner yleftlowercorner floats The xy position of the lowerleft corner in degrees

gridsize float The spacing between points of the grid in degrees

Notes

This dataset represents the geographic distribution of species The dataset is provided by Phillips et al 2006

The two species are

- "Bradypus variegatus" the Brownthroated Sloth
- "Microryzomys minutus" also known as the Forest Small Rice Rat a rodent that lives in Peru Colombia Ecuador Peru and Venezuela

- For an example of using this dataset with scikitlearn see examples

applicationsplotspeciesdistributionmodelingpy

References

- "Maximum entropy modeling of species geographic distributions" S J Phillips R P Anderson R E Schapire Ecological Modelling 190231259 2006

Examples using sklearn.datasets.fetch_species_distributions

- Species distribution modeling

- Kernel Density Estimate of Species Distributions

1590 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearn.datasets.get_data_home

sklearn.datasets.get_data_home(data_home=None)

Return the path of the scikitlearn data dir

This folder is used by some large dataset loaders to avoid downloading the data several times

By default the data dir is set to a folder named 'scikitlearn_data' in the user home folder

Alternatively it can be set by the 'SCIKIT_LEARN_DATA' environment variable or programmatically by giving an explicit folder path. The '~' symbol is expanded to the user home folder

If the folder does not already exist it is automatically created

Parameters

data_home: str, None. The path to scikitlearn data dir

Examples using sklearn.datasets.get_data_home

- Out-of-core classification of text documents

sklearn.datasets.load_boston

sklearn.datasets.load_boston(return_Xy=False)

Load and return the boston house prices dataset (regression)

Samples total: 506

Dimensionality: 13

Features: real, positive

Targets: real, 5 - 50

Read more in the User Guide

Parameters

return_Xy: boolean, default=False. If True, returns data, target instead of a Bunch object. See below for more information about the data and target object

New in version 0.18

Returns

data: Bunch. Dictionary-like object. The interesting attributes are 'data' the data to learn, 'target' the regression targets, 'DESCR' the full description of the dataset and 'filename' the physical location of boston.csv dataset. Added in version 0.20

data, target: tuple. If return_Xy is True. New in version 0.18

Notes

Changed in version 0.20: Fixed a wrong data point at 445.0

68 sklearn.datasets. Datasets: 1591

scikitlearn user guide Release 0213

Examples

from sklearn.datasets import load_boston

boston = load_boston

print(boston.data.shape)

506 13

Examples using sklearn.datasets.load_boston

- Outlier detection on a real data set
- Model Complexity Influence
- Effect of transforming the targets in regression model
- Plot individual and voting regression predictions
- Gradient Boosting regression
- Feature selection using SelectFromModel and LassoCV
- Imputing missing values before building an estimator
- Plotting CrossValidated Predictions

sklearn.datasets.load_breast_cancer

sklearn.datasets.load_breast_cancer(return_X_y=False)

Load and return the breast cancer wisconsin dataset classification

The breast cancer dataset is a classic and very easy binary classification dataset

Classes 2

Samples per class 212 357

Samples total 569

Dimensionality 30

Features real positive

Read more in the User Guide

Parameters

return_X_y boolean default False If True returns data target instead of a Bunch

object See below for more information about the data and target object

New in version 0.18

Returns

data Bunch Dictionary-like object the interesting attributes are 'data' the data to learn
'target' the classification labels 'target_names' the meaning of the labels 'feature_names'
the meaning of the features and 'DESCR' the full description of the dataset 'filename' the
physical location of breast cancer csv dataset added in version 0.20

data target tuple if return_X_y is True New in version 0.18

The copy of UCI ML Breast Cancer Wisconsin Diagnostic dataset is

downloaded from

1592 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

<https://googleusercontent.com/U2Uwz2>

Examples

Let's say you are interested in the samples 10 50 and 85 and want to know their class name

```
from sklearn.datasets import loadbreastcancer
```

```
data = loadbreastcancer
```

```
data.target[10:50:85]
```

```
array([1, 0, ...])
```

```
list(data.target_names)
```

```
['malignant', 'benign']
```

```
sklearn.datasets.load_diabetes
```

```
sklearn.datasets.load_diabetes(return_Xy=False)
```

Load and return the diabetes dataset regression

Samples total 442

Dimensionality 10

Features real 2 x 2

Targets integer 25 346

Read more in the User Guide

Parameters

return_Xy boolean default False If True returns data target instead of a Bunch

object See below for more information about the data and target object

New in version 0.18

Returns

data Bunch Dictionary-like object the interesting attributes are 'data' the data to learn 'target' the regression target for each sample 'datafilename' the physical location of diabetes data csv dataset and 'targetfilename' the physical location of diabetes targets csv dataset added in version 0.20

data target tuple if return_Xy is True New in version 0.18

Examples using sklearn.datasets.load_diabetes

- Cross-validation on diabetes Dataset Exercise
 - Imputing missing values before building an estimator
 - Lasso path using LARS
 - Linear Regression Example
 - Sparsity Example Fitting only features 1 and 2
 - Lasso and Elastic Net
 - Lasso model selection CrossValidation AIC BIC
- 68sklearn.datasets Datasets 1593

scikitlearn user guide Release 0213

sklearn.datasets.load_digits

sklearn.datasets.load_digits nclass=10 return_Xy=False

Load and return the digits dataset classification

Each datapoint is a 8x8 image of a digit

Classes 10

Samples per class 180

Samples total 1797

Dimensionality 64

Features integers 016

Read more in the User Guide

Parameters

nclass integer between 0 and 10 optional default=10 The number of classes to return

return_Xy boolean default=False If True returns data target instead of a Bunch

object See below for more information about the data and target object

New in version 0.18

Returns

data Bunch Dictionary-like object the interesting attributes are 'data' the data to learn

'images' the images corresponding to each sample 'target' the classification labels for

each sample 'target_names' the meaning of the labels and 'DESCR' the full description

of the dataset

data target tuple if return_Xy is True New in version 0.18

This is a copy of the test set of the UCI ML handwritten digits datasets

<https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits>

Examples

To load the data and visualize the images

```
from sklearn.datasets import load_digits
digits = load_digits
print(digits.data.shape)
1797 64
import matplotlib.pyplot as plt
plt.gray()
plt.matshow(digits.images[0])
plt.show
```

Examples using sklearn.datasets.load_digits

- The Johnson-Lindenstrauss bound for embedding with random projections
- Explicit feature map approximation for RBF (Radial Basis Function) kernels
- Recognizing handwritten digits

1594 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- Feature agglomeration
- Various Agglomerative Clustering on a 2D embedding of digits
- A demo of KMeans clustering on the handwritten digits data
- Pipelining chaining a PCA (Principal Component Analysis) and a logistic regression
- Selecting dimensionality reduction with Pipeline and GridSearchCV
- The Digit Dataset
- Early stopping of Gradient Boosting
- Digits Classification Exercise
- Crossvalidation on Digits Dataset Exercise
- Recursive feature elimination
- Comparing various online solvers
- L1 Penalty and Sparsity in Logistic Regression
- Manifold learning on handwritten digits Locally Linear Embedding Isomap
- Plotting Validation Curves
- Parameter estimation using grid search with crossvalidation
- Comparing randomized search and grid search for hyperparameter estimation
- Balance model complexity and crossvalidated score
- Plotting Learning Curves
- Kernel Density Estimation
- Dimensionality Reduction with Neighborhood Components Analysis
- Restricted Boltzmann Machine features for digit classification
- Compare Stochastic learning strategies for MLPClassifier
- Label Propagation digits Demonstrating performance
- Label Propagation digits active learning

sklearn.datasets loadfiles

sklearn.datasets loadfiles containerpath descriptionNone categoriesNone

loadcontentTrue shuffleTrue encodingNone de

codeerror'strict' randomstate0

Load text files with categories as subfolder names

Individual samples are assumed to be files stored a two levels folder structure such as the following

containerfolder

category1folder file1.txt file2.txt file42.txt

category2folder file43.txt file44.txt

The folder names are used as supervised signal label names The individual file names are not important

This function does not try to extract features into a numpy array or scipy sparse matrix In addition if

loadcontent is false it does not try to load the files in memory

68sklearn.datasets Datasets 1595

scikitlearn user guide Release 0213

To use text files in a scikitlearn classification or clustering algorithm you will need to use the sklearn featureextractiontext module to build a feature extraction transformer that suits your problem
If you set loadcontentTrue you should also specify the encoding of the text using the ‘encoding’ parameter
For many modern text files ‘utf8’ will be the correct encoding If you leave encoding equal to None then the content will be made of bytes instead of Unicode and you will not be able to use most functions in sklearn featureextractiontext
Similar feature extractors should be built for other kind of unstructured data input such as images audio video

Read more in the User Guide

Parameters
containerpath string or unicode Path to the main folder holding one subfolder per category
description string or unicode optional defaultNone A paragraph describing the characteristic of the dataset its source reference etc
categories A collection of strings or None optional defaultNone If None default load all the categories If not None list of category names to load other categories ignored
loadcontent boolean optional defaultTrue Whether to load or not the content of the different files If true a ‘data’ attribute containing the text information is present in the data structure returned If not a filenames attribute gives the path to the files
shuffle bool optional defaultTrue Whether or not to shuffle the data might be important for models that make the assumption that the samples are independent and identically distributed iid such as stochastic gradient descent
encoding string or None default is None If None do not try to decode the content of the files eg for images or other nontext content If not None encoding to use to decode text files to Unicode if loadcontent is True
decodeerror ‘strict’ ‘ignore’ ‘replace’ optional Instruction on what to do if a byte sequence is given to analyze that contains characters not of the given encoding Passed as keyword argument ‘errors’ to bytesdecode
randomstate int RandomState instance or None default0 Determines random number generation for dataset shuffling Pass an int for reproducible output across multiple function calls See Glossary
Returns
data Bunch Dictionarylike object the interesting attributes are either data the raw text data to learn or ‘filenames’ the files holding it ‘target’ the classification labels integer index ‘targetnames’ the meaning of the labels and ‘DESCR’ the full description of the dataset
sklearn.datasets.loadiris
sklearn.datasets.loadiris returnXyFalse
Load and return the iris dataset classification
The iris dataset is a classic and very easy multiclass classification dataset
1596 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Classes 3

Samples per class 50

Samples total 150

Dimensionality 4

Features real positive

Read more in the User Guide

Parameters

returnXy boolean defaultFalse If True returns data target instead of a Bunch

object See below for more information about the data andtarget object

New in version 018

Returns

data Bunch Dictionarylike object the interesting attributes are 'data' the data to learn
'target' the classification labels 'targetnames' the meaning of the labels 'featurenames'
the meaning of the features 'DESCR' the full description of the dataset 'filename' the
physical location of iris csv dataset added in version 020
data target tuple ifreturnXy is True New in version 018

Notes

Changed in version 020 Fixed two wrong data points according to Fisher's paper The new version is the same
as in R but not as in the UCI Machine Learning Repository

Examples

Let's say you are interested in the samples 10 25 and 50 and want to know their class name

from sklearn.datasets import loadiris

data loadiris

datatarget10 25 50

array0 0 1

listdatatargetnames

setosa versicolor virginica

Examples using sklearn.datasets.loadiris

- Plot classification probability
 - Kmeans Clustering
 - Concatenating multiple feature extraction methods
 - The Iris Dataset
 - PCA (Principal Component Analysis) example with Iris Dataset
 - Incremental PCA (Principal Component Analysis)
 - Comparison of LDA and PCA (Principal Component Analysis) 2D projection of Iris dataset
 - Plot the decision boundaries of a VotingClassifier
- 68sklearn.datasets Datasets 1597

scikitlearn user guide Release 0213

- Early stopping of Gradient Boosting
 - Plot the decision surfaces of ensembles of trees on the iris dataset
 - SVM (Support Vector Machine) Exercise
 - Test with permutations the significance of a classification score
 - Univariate Feature Selection
 - Gaussian process classification GPC on iris dataset
 - Regularization path of L1 Logistic Regression
 - Logistic Regression 3class Classifier
 - Plot multiclass SGD on the iris dataset
 - GMM covariances
 - Receiver Operating Characteristic ROC with cross validation
 - Nested versus nonnested crossvalidation
 - Confusion matrix
 - Receiver Operating Characteristic ROC
 - PrecisionRecall
 - Nearest Neighbors Classification
 - Nearest Centroid Classification
 - Comparing Nearest Neighbors with and without Neighborhood Components Analysis
 - Compare Stochastic learning strategies for MLPClassifier
 - Decision boundary of label propagation versus SVM (Support Vector Machine) on the Iris dataset
 - SVM (Support Vector Machine) with custom kernel
 - SVMAnova SVM (Support Vector Machine) with univariate feature selection
 - Plot different SVM (Support Vector Machine) classifiers in the iris dataset
 - RBF (Radial Basis Function) SVM (Support Vector Machine) parameters
 - Plot the decision surface of a decision tree on the iris dataset
 - Understanding the decision tree structure
- sklearn.datasets loadlinnerud
sklearn.datasets loadlinnerud returnXyFalse
Load and return the linnerud dataset multivariate regression
Samples total 20
Dimensionality 3 for both data and target
Features integer
Targets integer
Read more in the User Guide
Parameters
1598 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

returnXy boolean defaultFalse If True returns data target instead of a Bunch
object See below for more information about the data andtarget object
New in version 018

Returns
data Bunch Dictionarylike object the interesting attributes are 'data' and 'target' the two
multivariate datasets with 'data' corresponding to the exercise and 'target' corresponding
to the physiological measurements as well as 'featurenames' and 'targetnames' In addi
tion you will also have access to 'datafilename' the physical location of linnerud data csv
dataset and 'targetfilename' the physical location of linnerud targets csv dataset added
in version020

data target tuple ifreturnXy is True New in version 018

sklearndatasets loadsampleimage

sklearndatasets loadsampleimage imagename

Load the numpy array of a single sample image

Read more in the User Guide

Parameters

imagename chinajpg flowerjpg The name of the sample image loaded

Returns

img 3D array The image as a numpy array height x width x color

Examples

from sklearndatasets import loadsampleimage

china loadsampleimagechinajpg

chinadtype

dtypeuint8

chinashape

427 640 3

flower loadsampleimageflowerjpg

flowerdtype

dtypeuint8

flowershape

427 640 3

Examples using sklearndatasetsloadsampleimage

•Color Quantization using KMeans

sklearndatasets loadsampleimages

sklearndatasets loadsampleimages

Load sample images for image manipulation

Loads both china andflower

68sklearndatasets Datasets 1599

scikitlearn user guide Release 0213

Read more in the User Guide

Returns

data Bunch Dictionarylike object with the following attributes ‘images’ the two sample images ‘filenames’ the file names for the images and ‘DESCR’ the full description of the dataset

Examples

To load the data and visualize the images

from sklearn.datasets import load_sample_images

dataset = load_sample_images

load_dataset_images

2

first_img_data = dataset.images[0]

first_img_data.shape

(427, 640, 3)

first_img_data.dtype

dtype: uint8

sklearn.datasets.load_svmlight_file

sklearn.datasets.load_svmlight_file fn: features=None dtype: class ‘numpy.float64’

multi_label=False zero_based=‘auto’ query_id=False

offset=0 length=1

Load datasets in the svmlight / libsvm format into sparse CSR (Compressed Sparse Row) matrix

This format is a text-based format with one sample per line. It does not store zero-valued features hence is

suitable for sparse dataset

The first element of each line can be used to store a target variable to predict

This format is used as the default format for both svmlight and the libsvm command line programs

Parsing a text-based source can be expensive. When working on repeatedly on the same dataset it is recom-

mended to wrap this loader with joblib.MemoryCache to store a memmapped backup of the CSR (Compressed Sparse Row) results of the

first call and benefit from the near instantaneous loading of memmapped structures for the subsequent calls

In case the file contains a pairwise preference constraint known as “qid” in the svmlight format these are

ignored unless the query_id parameter is set to True. These pairwise preference constraints can be used to

constraint the combination of samples when using pairwise loss functions as is the case in some learning to

rank problems so that only pairs with the same query_id value are considered

This implementation is written in Cython and is reasonably fast. However a faster API-compatible loader is

also available at

https://github.com/blondel/svmlight_loader

Parameters

fn: file-like int Path to a file to load. If a path ends in “gz” or “bz2” it will be uncom-

pressed on the fly. If an integer is passed it is assumed to be a file descriptor. A file-like

or file descriptor will not be closed by this function. A file-like object must be opened in

binary mode

1600 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

nfeatures int or None The number of features to use If None it will be inferred This argument is useful to load several files that are subsets of a bigger sliced dataset each subset might not have examples of every feature hence the inferred shape might vary from one slice to another nfeatures is only required if offset orlength are passed a non default value

dtype numpy data type default npfloat64 Data type of dataset to be loaded This will be the data type of the output numpy arrays Xandy

multilabel boolean optional default False Samples may have several labels each see <https://www.cs.cmu.edu/~dwj/libsvmtools/datasets/multilabel.html>

zerobased boolean or "auto" optional default "auto" Whether column indices in f are zero based True or onebased False If column indices are onebased they are transformed to zerobased to match PythonNumPy conventions If set to "auto" a heuristic check is applied to determine this from the file contents Both kinds of files occur "in the wild" but they are unfortunately not selfidentifying Using "auto" or True should always be safe when nooffset orlength is passed If offset orlength are passed the "auto" mode falls back tozerobasedTrue to avoid having the heuristic check yield inconsistent results on different segments of the file

queryid boolean default False If True will return the queryid array for each file

offset integer optional default 0 Ignore the offset first bytes by seeking forward then discarding the following bytes up until the next new line character

length integer optional default 1 If strictly positive stop reading any new line of data once the position in the file has reached the offset length bytes threshold

Returns
Xscipy sparse matrix of shape nsamples nfeatures
y ndarray of shape nsamples or in the multilabel a list of tuples of length nsamples
queryid array of shape nsamples queryid for each sample Only returned when queryid is set to True
See also

loadsvmlightfiles similar function for loading multiple files in this format enforcing the same number of featurescolumns on all of them

Examples
To use joblibMemory to cache the svmlight file

```
from joblib import Memory
from datasets import loadsvmlightfile
mem = Memory(cache)
memcache
defgetdata
data loadsvmlightfilemysvmlightfile
returndata0 data1
X y getdata
```

68sklearn datasets Datasets 1601

scikitlearn user guide Release 0213

sklearn.datasets.loadsvmlightfiles

sklearn.datasets.loadsvmlightfiles files nfeaturesNone dtypeclass

'numpy.float64' multilabelFalse

zerobased'auto' queryidFalse offset0 length

1

Load dataset from multiple files in SVMlight format

This function is equivalent to mapping loadsvmlightfile over a list of files except that the results are concatenated into a single flat list and the samples vectors are constrained to all have the same number of features. In case the file contains a pairwise preference constraint known as "qid" in the svmlight format these are ignored unless the queryid parameter is set to True. These pairwise preference constraints can be used to constraint the combination of samples when using pairwise loss functions as is the case in some learning to rank problems so that only pairs with the same queryid value are considered.

Parameters

files iterable over str filelike int Paths of files to load. If a path ends in "gz" or "bz2" it will be uncompressed on the fly. If an integer is passed it is assumed to be a file descriptor. Filelikes and file descriptors will not be closed by this function. Filelike objects must be

opened in binary mode.

nfeatures int or None The number of features to use. If None it will be inferred from the maximum column index occurring in any of the files.

This can be set to a higher value than the actual number of features in any of the input files but setting it to a lower value will cause an exception to be raised.

dtype numpy data type default np.float64 Data type of dataset to be loaded. This will be the data type of the output numpy arrays.

Xandy multilabel boolean optional Samples may have several labels each see <https://www.cs.cmu.edu/~tgcj/libsvmtools/datasets/multilabel.html>

zerobased boolean or "auto" optional Whether column indices in f are zerobased. True or onebased. False. If column indices are onebased they are transformed to zerobased to match Python/NumPy conventions. If set to "auto" a heuristic check is applied to determine this from the file contents. Both kinds of files occur "in the wild" but they are unfortunately not self-identifying. Using "auto" or True should always be safe when no offset or length is passed. If offset or length are passed the "auto" mode falls back to zerobased=True to avoid having the heuristic check yield inconsistent results on different segments of the file.

queryid boolean defaults to False. If True will return the queryid array for each file.

offset integer optional default 0 Ignore the offset first bytes by seeking forward then discarding the following bytes up until the next new line character.

length integer optional default 1 If strictly positive stop reading any new line of data once the position in the file has reached the offset + length bytes threshold.

Returns

X1 y1 ... Xn yn

where each Xi yi pair is the result from loadsvmlightfile(files[i])

If queryid is set to True this will return instead X1 y1 q1

Xn yn qn where Xi yi qi is the result from

loadsvmlightfile(files[i])

See also

1602 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

loadsvmlightfile

Notes

When fitting a model to a matrix Xtrain and evaluating it against a matrix Xtest it is essential that Xtrain and Xtest have the same number of features Xtrainshape1 Xtestshape1 This may not be the case if you load the files individually with loadsvmlightfile

sklearndatasets loadwine

sklearndatasets loadwine returnXyFalse

Load and return the wine dataset classification

New in version 018

The wine dataset is a classic and very easy multiclass classification dataset

Classes 3

Samples per class 597148

Samples total 178

Dimensionality 13

Features real positive

Read more in the User Guide

Parameters

returnXy boolean defaultFalse If True returns data target instead of a Bunch

object See below for more information about the data andtarget object

Returns

data Bunch Dictionarylike object the interesting attributes are 'data' the data to learn

'target' the classification labels 'targetnames' the meaning of the labels 'featurenames'

the meaning of the features and 'DESCR' the full description of the dataset

data target tuple ifreturnXy is True

The copy of UCI ML Wine Data Set dataset is downloaded and modified to fit

standard format from

<https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data>

Examples

Let's say you are interested in the samples 10 80 and 140 and want to know their class name

from sklearndatasets import loadwine

data loadwine

datatarget10 80 140

array0 1 2

listdatatargetnames

class0 class1 class2

68sklearndatasets Datasets 1603

scikitlearn user guide Release 0213

Examples using sklearn.datasets.loadwine

- Importance of Feature Scaling

682 Samples generator

datasets.make_biclusters(shape, nclusters) Generate an array with constant block diagonal structure for biclustering

datasets.make_blobs(nsamples, nfeatures) Generate isotropic Gaussian blobs for clustering

datasets.make_checkerboard(shape, nclusters) Generate an array with block checkerboard structure for bi clustering

datasets.make_circles(nsamples, shuffle) Make a large circle containing a smaller circle in 2d

datasets.make_classification(nsamples)

Generate a random nclass classification problem

datasets.make_friedman1(nsamples) Generate the “Friedman 1” regression problem

datasets.make_friedman2(nsamples, noise)

Generate the “Friedman 2” regression problem

datasets.make_friedman3(nsamples, noise)

Generate the “Friedman 3” regression problem

datasets.make_gaussian_quantiles(mean)

Generate isotropic Gaussian and label samples by quantile

datasets.make_hastie102(nsamples) Generates data for binary classification used in Hastie et al

datasets.make_low_rank_matrix(nsamples)

Generate a mostly low rank matrix with bellshaped singular values

datasets.make_moons(nsamples, shuffle) Make two interleaving half circles

datasets.make_multilabel_classification() Generate a random multilabel classification problem

datasets.make_regression(nsamples) Generate a random regression problem

datasets.make_s_curve(nsamples, noise) Generate an S curve dataset

datasets.make_sparse_coded_signal(nsamples)

Generate a signal as a sparse combination of dictionary elements

datasets.make_sparse_posdef_matrix(dim) Generate a sparse symmetric definite positive matrix

datasets.make_sparse_uncorrelated() Generate a random regression problem with sparse uncorrelated design

datasets.make_spd_matrix(ndim, randomstate) Generate a random symmetric positive definite matrix

datasets.make_swiss_roll(nsamples, noise)

Generate a swiss roll dataset

sklearn.datasets.make_biclusters(shape, nclusters, noise=0.0, min_val=10, max_val=100, shuffle=True, randomstate=None)

Generate an array with constant block diagonal structure for biclustering

Read more in the User Guide

Parameters

shape iterable n rows, n cols The shape of the result

nclusters integer The number of biclusters

noise float optional default=0.0 The standard deviation of the gaussian noise

1604 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

minval int optional default10 Minimum value of a bicluster

maxval int optional default100 Maximum value of a bicluster

shuffle boolean optional defaultTrue Shuffle the samples

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape shape The generated array

rows array of shape nclusters Xshape0 The indicators for cluster membership of each row

cols array of shape nclusters Xshape1 The indicators for cluster membership of each column

See also

makecheckerboard

References

1

Examples using sklearndatasetsmakebiclusters

•A demo of the Spectral CoClustering algorithm

sklearndatasets makeblobs

sklearndatasets makeblobs nsamples100 nfeatures2 centersNone clusterstd10

centerbox100 100 shuffleTrue randomstateNone

Generate isotropic Gaussian blobs for clustering

Read more in the User Guide

Parameters

nsamples int or arraylike optional default100 If int it is the total number of points

equally divided among clusters If arraylike each element of the sequence indicates the number of samples per cluster

nfeatures int optional default2 The number of features for each sample

centers int or array of shape ncenters nfeatures optional defaultNone The number of centers to generate or the fixed center locations If nsamples is an int and centers is None

3 centers are generated If nsamples is arraylike centers must be either None or an array of length equal to the length of nsamples

clusterstd float or sequence of floats optional default10 The standard deviation of the clusters

centerbox pair of floats min max optional default100 100 The bounding box for each cluster center when centers are generated at random

68sklearndatasets Datasets 1605

scikitlearn user guide Release 0213

shuffle boolean optional default True Shuffle the samples

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

X array of shape (nsamples, nfeatures) The generated samples

y array of shape (nsamples) The integer labels for cluster membership of each sample

See also

makeclassification a more intricate variant

Examples

```
from sklearn.datasets.samples_generator import make_blobs
X, y = make_blobs(n_samples=10, centers=3, n_features=2,
                  random_state=0)
print(X.shape)
10 2
y
array([0, 0, 1, 0, 2, 2, 2, 1, 1, 0])
X, y = make_blobs(n_samples=3, centers=3, n_features=2,
                  random_state=0)
print(X.shape)
10 2
y
array([0, 1, 2, 0, 2, 2, 2, 1, 1, 0])
```

Examples using sklearn.datasets.make_blobs

- Comparing anomaly detection algorithms for outlier detection on toy datasets
- Probability calibration of classifiers
- Probability Calibration for 3-class classification
- Normal and Shrinkage Linear Discriminant Analysis for classification
- A demo of the meanshift clustering algorithm
- Demonstration of kmeans assumptions
- Demo of affinity propagation clustering algorithm
- Demo of DBSCAN clustering algorithm
- Inductive Clustering
- Compare BIRCH and MiniBatchKMeans
- Comparison of the KMeans and MiniBatchKMeans clustering algorithms
- Comparing different hierarchical linkage methods on toy datasets
- Selecting the number of clusters with silhouette analysis on KMeans clustering
- Comparing different clustering algorithms on toy datasets

1606 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- Plot randomly generated classification dataset
- SGD Maximum margin separating hyperplane
- Plot multinomial and OnevsRest Logistic Regression
- Demonstrating the different strategies of KBinsDiscretizer
- SVM (Support Vector Machine) Maximum margin separating hyperplane
- SVM (Support Vector Machine) Separating hyperplane for unbalanced classes

sklearndatasets makecheckerboard
sklearndatasets makecheckerboard shape nclusters noise00 minval10 maxval100

shuffleTrue randomstateNone

Generate an array with block checkerboard structure for biclustering

Read more in the User Guide

Parameters

shape iterable nrows ncols The shape of the result

nclusters integer or iterable nrowclusters ncolumnclusters The number of row and column clusters

noise float optional default00 The standard deviation of the gaussian noise

minval int optional default10 Minimum value of a bicluster

maxval int optional default100 Maximum value of a bicluster

shuffle boolean optional defaultTrue Shuffle the samples

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape shape The generated array

rows array of shape nclusters Xshape0 The indicators for cluster membership of each row

cols array of shape nclusters Xshape1 The indicators for cluster membership of each column

See also

makebicclusters

References

1

Examples using sklearndatasetsmakecheckerboard

- A demo of the Spectral Biclustering algorithm

68sklearndatasets Datasets 1607

scikitlearn user guide Release 0213

sklearn.datasets.make_circles

sklearn.datasets.make_circles nsamples=100 shuffle=True noise=None random_state=None
factor=0.8

Make a large circle containing a smaller circle in 2d

A simple toy dataset to visualize clustering and classification algorithms

Read more in the User Guide

Parameters

nsamples int optional default=100 The total number of points generated. If odd the inner
circle will have one point more than the outer circle

shuffle bool optional default=True Whether to shuffle the samples

noise double or None default=None Standard deviation of Gaussian noise added to the data

random_state int RandomState instance or None default=None Determines random number gen-
eration for dataset shuffling and noise. Pass an int for reproducible output across multiple
function calls. See Glossary

factor float double 1 default=0.8 Scale factor between inner and outer circle

Returns

X array of shape (nsamples, 2) The generated samples

y array of shape (nsamples,) The integer labels 0 or 1 for class membership of each sample

Examples using sklearn.datasets.make_circles

- Classifier comparison
- Comparing different hierarchical linkage methods on toy datasets
- Comparing different clustering algorithms on toy datasets
- Kernel PCA (Principal Component Analysis)
- Hashing feature transformation using Totally Random Trees
- tSNE The effect of various perplexity values on the shape
- Varying regularization in Multilayer Perceptron
- Compare Stochastic learning strategies for MLPClassifier
- Feature discretization
- Label Propagation learning a complex structure

sklearn.datasets.make_classification

sklearn.datasets.make_classification nsamples=100 nfeatures=20 n_informative=2
n_redundant=2 n_repeated=0 n_classes=2

n_clusters_per_class=2 weights=None flipy=0.01

class_sep=10 hypercube=True shift=0.0 scale=1.0

shuffle=True random_state=None

Generate a random nclass classification problem

1608 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

This initially creates clusters of points normally distributed std1 about vertices of an ninformative dimensional hypercube with sides of length 2classep and assigns an equal number of clusters to each class It introduces interdependence between these features and adds various types of further noise to the data Without shuffling Xhorizontally stacks features in the following order the primary ninformative features followed by nredundant linear combinations of the informative features followed by nrepeated duplicates drawn randomly with replacement from the informative and redundant features The remaining features are filled with random noise Thus without shuffling all useful features are contained in the columns

X ninformative nredundant nrepeated

Read more in the User Guide

Parameters

nsamples int optional default100 The number of samples

nfeatures int optional default20 The total number of features These comprise ninformative informative features nredundant redundant features nrepeated duplicated features and nfeaturesninformativenredundantnrepeated useless features drawn at random

ninformative int optional default2 The number of informative features Each class is composed of a number of gaussian clusters each located around the vertices of a hypercube in a subspace of dimension ninformative For each cluster informative features are drawn independently from N0 1 and then randomly linearly combined within each cluster in order to add covariance The clusters are then placed on the vertices of the hypercube nredundant int optional default2 The number of redundant features These features are generated as random linear combinations of the informative features nrepeated int optional default0 The number of duplicated features drawn randomly from the informative and the redundant features nclasses int optional default2 The number of classes or labels of the classification problem

nclustersperclass int optional default2 The number of clusters per class

weights list of floats or None defaultNone The proportions of samples assigned to each class If None then classes are balanced Note that if lenweights nclasses

1 then the last class weight is automatically inferred More than nsamples samples may be returned if the sum of weights exceeds 1

flipy float optional default001 The fraction of samples whose class are randomly exchanged Larger values introduce noise in the labels and make the classification task harder

classep float optional default10 The factor multiplying the hypercube size Larger values spread out the clustersclasses and make the classification task easier

hypercube boolean optional defaultTrue If True the clusters are put on the vertices of a hypercube If False the clusters are put on the vertices of a random polytope

shift float array of shape nfeatures or None optional default00 Shift features by the specified value If None then features are shifted by a random value drawn in classep classep

scale float array of shape nfeatures or None optional default10 Multiply features by the specified value If None then features are scaled by a random value drawn in 1 100

Note that scaling happens after shifting

shuffle boolean optional defaultTrue Shuffle the samples and the features

68sklearn datasets Datasets 1609

scikitlearn user guide Release 0213

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape nsamples nfeatures The generated samples

yarray of shape nsamples The integer labels for class membership of each sample

See also

makeblobs simplified variant

makemultilabelclassification unrelated generator for multilabel tasks

Notes

The algorithm is adapted from Guyon 1 and was designed to generate the “Madelon” dataset

References

1

Examples using sklearn.datasets.makeclassification

- Comparison of Calibration of Classifiers
- Probability Calibration curves
- Classifier comparison
- Plot randomly generated classification dataset
- Feature importances with forests of trees
- OOB Errors for Random Forests
- Feature transformations with ensembles of trees
- Pipeline Anova SVM (Support Vector Machine)
- Recursive feature elimination with crossvalidation
- Neighborhood Components Analysis Illustration
- Varying regularization in Multilayer Perceptron
- Feature discretization
- Scaling the regularization parameter for SVCs

sklearn.datasets.makefriedman1

sklearn.datasets.makefriedman1 nsamples100 nfeatures10 noise00 randomstateNone

Generate the “Friedman 1” regression problem

This dataset is described in Friedman 1 and Breiman 2

1610 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

InputsXare independent features uniformly distributed on the interval 0 1 The output yis created according to the formula

$$yX = 10 \sin(\pi X_0 X_1 - 20 X_2^2 - 0.5 X_3 - 10 X_4)$$

↪3 5X 4 noise N0 1

Out of the nfeatures features only 5 are actually used to compute y The remaining features are independent of y

The number of features has to be 5

Read more in the User Guide

Parameters

nsamples int optional default100 The number of samples

nfeatures int optional default10 The number of features Should be at least 5

noise float optional default00 The standard deviation of the gaussian noise applied to the

output

randomstate int RandomState instance or None default Determines random number generation for dataset noise Pass an int for reproducible output across multiple function calls

SeeGlossary

Returns

Xarray of shape nsamples nfeatures The input samples

yarray of shape nsamples The output values

References

12

sklearndatasets makefriedman2

sklearndatasets makefriedman2 nsamples100 noise00 randomstateNone

Generate the “Friedman 2” regression problem

This dataset is described in Friedman 1 and Breiman 2

InputsXare 4 independent features uniformly distributed on the intervals

$$0 \leq X_0 \leq 100$$

$$40\pi \leq X_1 \leq 560\pi$$

$$0 \leq X_2 \leq 1$$

$$1 \leq X_3 \leq 11$$

The output yis created according to the formula

$$yX = X_0^2 - X_1 X_2 + 1 - X_1 X_3 + 20$$

↪5 noise N0 1

Read more in the User Guide

Parameters

nsamples int optional default100 The number of samples

68sklearndatasets Datasets 1611

scikitlearn user guide Release 0213

noise float optional default00 The standard deviation of the gaussian noise applied to the output

randomstate int RandomState instance or None default Determines random number generation for dataset noise Pass an int for reproducible output across multiple function calls SeeGlossary

Returns

Xarray of shape nsamples 4 The input samples

yarray of shape nsamples The output values

References

12

sklearndatasets makefriedman3

sklearndatasets makefriedman3 nsamples100 noise00 randomstateNone

Generate the “Friedman 3” regression problem

This dataset is described in Friedman 1 and Breiman 2

InputsXare 4 independent features uniformly distributed on the intervals

0 X 0 100

40pi X 1 560 pi

0 X 2 1

1 X 3 11

The output yis created according to the formula

$y = X_0 \arctan(X_1 X_2 - 1 X_1 X_3 X_0) + \text{noise}$

$\rightarrow N(0, 1)$

Read more in the User Guide

Parameters

nsamples int optional default100 The number of samples

noise float optional default00 The standard deviation of the gaussian noise applied to the output

randomstate int RandomState instance or None default Determines random number generation for dataset noise Pass an int for reproducible output across multiple function calls SeeGlossary

Returns

Xarray of shape nsamples 4 The input samples

yarray of shape nsamples The output values

References

12

1612 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearn.datasets.makegaussianquantiles

sklearn.datasets.makegaussianquantiles mean=None cov=10 nsamples=100

nfeatures=2 nclasses=3 shuffle=True

randomstate=None

Generate isotropic Gaussian and label samples by quantile

This classification dataset is constructed by taking a multidimensional standard normal distribution and defining classes separated by nested concentric multidimensional spheres such that roughly equal numbers of samples are in each class quantiles of the χ^2 distribution

Read more in the User Guide

Parameters

mean array of shape nfeatures optional default=None The mean of the multi-dimensional normal distribution If None then use the origin 0 0

cov float optional default=1 The covariance matrix will be this value times the unit matrix

This dataset only produces symmetric normal distributions

nsamples int optional default=100 The total number of points equally divided among classes

nfeatures int optional default=2 The number of features for each sample

nclasses int optional default=3 The number of classes

shuffle boolean optional default=True Shuffle the samples

randomstate int RandomState instance or None default=None Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

X array of shape nsamples nfeatures The generated samples

y array of shape nsamples The integer labels for quantile membership of each sample

Notes

The dataset is from Zhu et al 1

References

1

Examples using sklearn.datasets.makegaussianquantiles

- Plot randomly generated classification dataset
- Twoclass AdaBoost
- Multiclass AdaBoosted Decision Trees

68sklearn.datasets Datasets 1613

scikitlearn user guide Release 0213

sklearn.datasets.make_hastie102

sklearn.datasets.make_hastie102 nsamples=12000 randomstate=None

Generates data for binary classification used in Hastie et al 2009 Example 102

The ten features are standard independent Gaussian and the target y is defined by

$y_i = 1$ if $\sum_{j=1}^{10} x_{ij} \geq 0.5$ else $y_i = 0$

Read more in the User Guide

Parameters

nsamples int optional default=12000 The number of samples

randomstate int RandomState instance or None default=None Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

X array of shape (nsamples, 10) The input samples

y array of shape (nsamples,) The output values

See also

make_gaussian_quantiles a generalization of this dataset approach

References

1

Examples using sklearn.datasets.make_hastie102

- Gradient Boosting regularization
- Discrete versus Real AdaBoost
- Early stopping of Gradient Boosting
- Demonstration of multivariate evaluation on cross_val_score and GridSearchCV

sklearn.datasets.make_low_rank_matrix

sklearn.datasets.make_low_rank_matrix nsamples=100 nfeatures=100 effective_rank=10

tail_strength=0.5 randomstate=None

Generate a mostly low rank matrix with bell-shaped singular values

Most of the variance can be explained by a bell-shaped curve of width effective_rank the low rank part of the singular values profile is

$\frac{1}{i} \cdot \text{tail_strength} \cdot \exp(-i / \text{effective_rank})$

The remaining singular values' tail is fat decreasing as

$\frac{1}{i} \cdot \text{tail_strength} \cdot \exp(-i / \text{effective_rank})$

1614 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The low rank part of the profile can be considered the structured signal part of the data while the tail can be considered the noisy part of the data that cannot be summarized by a low number of linear components singular vectors

This kind of singular profiles is often seen in practice for instance

- gray level pictures of faces
- TFIDF vectors of text documents crawled from the web

Read more in the User Guide

Parameters

nsamples int optional default100 The number of samples
nfeatures int optional default100 The number of features
effectiverank int optional default10 The approximate number of singular vectors required to explain most of the data by linear combinations
tailstrength float between 00 and 10 optional default05 The relative importance of the fat noisy tail of the singular values profile
randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape nsamples nfeatures The matrix
sklearn.datasets.makemoons
sklearn.datasets.makemoons nsamples100 shuffleTrue noiseNone randomstateNone

Make two interleaving half circles

A simple toy dataset to visualize clustering and classification algorithms Read more in the User Guide

Parameters

nsamples int optional default100 The total number of points generated
shuffle bool optional defaultTrue Whether to shuffle the samples
noise double or None defaultNone Standard deviation of Gaussian noise added to the data
randomstate int RandomState instance or None default Determines random number generation for dataset shuffling and noise Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape nsamples 2 The generated samples
yarray of shape nsamples The integer labels 0 or 1 for class membership of each sample

Examples using sklearn.datasets.makemoons

- Comparing anomaly detection algorithms for outlier detection on toy datasets
- Classifier comparison
- Comparing different hierarchical linkage methods on toy datasets

68sklearn.datasets Datasets 1615

scikitlearn user guide Release 0213

- Comparing different clustering algorithms on toy datasets
- Varying regularization in Multilayer Perceptron
- Compare Stochastic learning strategies for MLPClassifier
- Feature discretization

sklearn.datasets.makemultilabelclassification

sklearn.datasets.makemultilabelclassification nsamples100 nfeatures20

nclasses5 nlabels2 length50

allowunlabeledTrue sparseFalse

returnindicator'dense' re

turnndistributionsFalse ran

domstateNone

Generate a random multilabel classification problem

For each sample the generative process is

- pick the number of labels $n \sim \text{Poisson}(n_{\text{labels}})$
- n times choose a class $c \sim \text{Multinomial}(\theta_c)$
- pick the document length $k \sim \text{Poisson}(length)$
- k times choose a word $w \sim \text{Multinomial}(\theta_{cw})$

In the above process rejection sampling is used to make sure that n is never zero or more than $nclasses$ and that the document length is never zero Likewise we reject classes which have already been chosen

Read more in the User Guide

Parameters

nsamples int optional default100 The number of samples

nfeatures int optional default20 The total number of features

nclasses int optional default5 The number of classes of the classification problem

nlabels int optional default2 The average number of labels per instance More precisely the number of labels per sample is drawn from a Poisson distribution with $nlabels$ as its expected value but samples are bounded using rejection sampling by $nclasses$ and must be nonzero if `allowunlabeled` is False

length int optional default50 The sum of the features number of words if documents is drawn from a Poisson distribution with this expected value

allowunlabeled bool optional defaultTrue If True some instances might not belong to any class

sparse bool optional defaultFalse If True return a sparse feature matrix

New in version 0.17 parameter to allow sparse output

returnindicator 'dense' default 'sparse' False If dense return Y in the dense binary

indicator format If sparse return Y in the sparse binary indicator format False

returns a list of lists of labels

turnndistributions bool optional defaultFalse If True return the prior class probability and conditional probabilities of features given classes from which the data was drawn

1616 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape nsamples nfeatures The generated samples

Yarray or sparse CSR (Compressed Sparse Row) matrix of shape nsamples nclasses The label sets

pc array shape nclasses The probability of each class being drawn Only returned if

returndistributionsTrue

pwc array shape nfeatures nclasses The probability of each feature being drawn given

each class Only returned if returndistributionsTrue

Examples using sklearn.datasets.makemultilabelclassification

- Multilabel classification

- Plot randomly generated multilabel dataset

sklearn.datasets.makeregression

sklearn.datasets.makeregression nsamples100 nfeatures100 ninformative10

ntargets1 bias00 effectiverankNone

tailstrength05 noise00 shuffleTrue coeffFalse

randomstateNone

Generate a random regression problem

The input set can either be well conditioned by default or have a low rankfat tail singular profile See

makelowrankmatrix for more details

The output is generated by applying a potentially biased random linear regression model with

ninformative nonzero regressors to the previously generated input and some gaussian centered noise with some adjustable scale

Read more in the User Guide

Parameters

nsamples int optional default100 The number of samples

nfeatures int optional default100 The number of features

ninformative int optional default10 The number of informative features ie the number of features used to build the linear model used to generate the output

ntargets int optional default1 The number of regression targets ie the dimension of the y output vector associated with a sample By default the output is a scalar

bias float optional default00 The bias term in the underlying linear model

effectiverank int or None optional defaultNone

if not None The approximate number of singular vectors required to explain most of the input data by linear combinations Using this kind of singular spectrum in the input allows

the generator to reproduce the correlations often observed in practice

if None The input set is well conditioned centered and gaussian with unit variance

68sklearn.datasets Datasets 1617

scikitlearn user guide Release 0213

tailstrength float between 00 and 10 optional default05 The relative importance of the fat noisy tail of the singular values profile if effectiverank is not None
noise float optional default00 The standard deviation of the gaussian noise applied to the output
shuffle boolean optional defaultTrue Shuffle the samples and the features
coef boolean optional defaultFalse If True the coefficients of the underlying linear model are returned
randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape nsamples nfeatures The input samples
yarray of shape nsamples or nsamples ntargets The output values
coef array of shape nfeatures or nfeatures ntargets optional The coefficient of the underlying linear model It is returned only if coef is True
Examples using sklearn.datasets.makeregression

- Prediction Latency
- Effect of transforming the targets in regression model
- Plot Ridge coefficients as a function of the L2 regularization
- Robust linear model estimation using RANSAC
- Lasso on dense and sparse data
- HuberRegressor vs Ridge on dataset with strong outliers

sklearn.datasets.makescurve
sklearn.datasets.makescurve nsamples100 noise00 randomstateNone
Generate an S curve dataset

Read more in the User Guide

Parameters

nsamples int optional default100 The number of sample points on the S curve
noise float optional default00 The standard deviation of the gaussian noise
randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape nsamples 3 The points
tarray of shape nsamples The univariate position of the sample according to the main dimension of the points in the manifold

scikitlearn user guide Release 0213

Examples using sklearn.datasets.make_s_curve

- tSNE The effect of various perplexity values on the shape
- Comparison of Manifold Learning methods

sklearn.datasets.make_sparse_coded_signal

sklearn.datasets.make_sparse_coded_signal n_samples n_components n_features

n_nonzero_coefs random_state=None

Generate a signal as a sparse combination of dictionary elements

Returns a matrix Y of shape (n_samples, n_features) such as D is (n_features, n_components) X is (n_components, n_samples) and each column of X has exactly n_nonzero_coefs nonzero elements

Read more in the User Guide

Parameters

n_samples int number of samples to generate

n_components int number of components in the dictionary

n_features int number of features of the dataset to generate

n_nonzero_coefs int number of active nonzero coefficients in each sample

random_state int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

data array of shape (n_samples, n_features) The encoded signal Y

dictionary array of shape (n_features, n_components) The dictionary with normalized components D

code array of shape (n_components, n_samples) The sparse code such that each column of this matrix has exactly n_nonzero_coefs nonzero items X

Examples using sklearn.datasets.make_sparse_coded_signal

- Orthogonal Matching Pursuit

sklearn.datasets.make_sparse_spd_matrix

sklearn.datasets.make_sparse_spd_matrix dim1 alpha=0.95 norm_diag=False

smallest_coef=0.1 largest_coef=0.9 random_state=None

Generate a sparse symmetric definite positive matrix

Read more in the User Guide

Parameters

dim integer optional default=1 The size of the random matrix to generate

alpha float between 0 and 1 optional default=0.95 The probability that a coefficient is zero see notes Larger values enforce more sparsity

68sklearn.datasets Datasets 1619

scikitlearn user guide Release 0213

normdiag boolean optional defaultFalse Whether to normalize the output matrix to make the leading diagonal elements all 1

smallestcoef float between 0 and 1 optional default01 The value of the smallest coefficient

largestcoef float between 0 and 1 optional default09 The value of the largest coefficient

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

prec sparse matrix of shape dim dim The generated matrix

See also

makespdmatrix

Notes

The sparsity is actually imposed on the cholesky factor of the matrix Thus alpha does not translate directly into the filling fraction of the matrix itself

Examples using sklearn.datasets.makesparsepdmatrix

•Sparse inverse covariance estimation

sklearn.datasets.makesparseuncorrelated

sklearn.datasets.makesparseuncorrelated nsamples100 nfeatures10 randomstateNone

Generate a random regression problem with sparse uncorrelated design

This dataset is described in Celeux et al 1 as

$X = N \times 1$

$y = X_0 + 2X_1 + 2X_2 + 15X_3$

Only the first 4 features are informative The remaining features are useless

Read more in the User Guide

Parameters

nsamples int optional default100 The number of samples

nfeatures int optional default10 The number of features

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

X array of shape nsamples nfeatures The input samples

y array of shape nsamples The output values

1620 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

References

1

sklearndatasets makespdmatrix

sklearndatasets makespdmatrix ndim randomstateNone

Generate a random symmetric positivedefinite matrix

Read more in the User Guide

Parameters

ndim int The matrix dimension

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape ndim ndim The random symmetric positivedefinite matrix

See also

makesparsespdmatrix

sklearndatasets makeswissroll

sklearndatasets makeswissroll nsamples100 noise00 randomstateNone

Generate a swiss roll dataset

Read more in the User Guide

Parameters

nsamples int optional default100 The number of sample points on the S curve

noise float optional default00 The standard deviation of the gaussian noise

randomstate int RandomState instance or None default Determines random number generation for dataset creation Pass an int for reproducible output across multiple function calls See Glossary

Returns

Xarray of shape nsamples 3 The points

tarray of shape nsamples The univariate position of the sample according to the main dimension of the points in the manifold

Notes

The algorithm is from Marsland 1

References

1

68sklearndatasets Datasets 1621

scikitlearn user guide Release 0213

Examples using sklearndatasetsmakeswissroll

- Hierarchical clustering structured vs unstructured ward
- Swiss Roll reduction with LLE

69sklearndecomposition Matrix Decomposition

The sklearndecomposition module includes matrix decomposition algorithms including among others PCA (Principal Component Analysis) NMF (Non-Negative Matrix Factorization) or ICA Most of the algorithms of this module can be regarded as dimensionality reduction techniques

User guide See the Decomposing signals in components matrix factorization problems section for further details

decompositionDictionaryLearning Dictionary learning

decompositionFactorAnalysis ncomponents

Factor Analysis FA

decompositionFastICA ncomponents FastICA a fast algorithm for Independent Component Analysis

decompositionIncrementalPCA ncomponents

Incremental principal components analysis IPCA

decompositionKernelPCA ncomponents Kernel Principal component analysis KPCA

decompositionLatentDirichletAllocation Latent Dirichlet Allocation with online variational Bayes algorithm

decompositionMiniBatchDictionaryLearning Minibatch dictionary learning

decompositionMiniBatchSparsePCA Minibatch Sparse Principal Components Analysis

decompositionNMF ncomponents init NonNegative Matrix Factorization NMF (Non-Negative Matrix Factorization)

decompositionPCA ncomponents copy Principal component analysis PCA (Principal Component Analysis)

decompositionSparsePCA ncomponents Sparse Principal Components Analysis SparsePCA

decompositionSparseCoder dictionary Sparse coding

decompositionTruncatedSVD ncomponents

Dimensionality reduction using truncated SVD aka LSA

691sklearndecomposition DictionaryLearning

classsklearndecomposition DictionaryLearning ncomponentsNone alpha1

maxiter1000 tol1e08

fitalgorithm'lasso' trans

formalgorithm'omp' trans

formnonzerocoefsNone trans

formalphaNone njobsNone

codeinitNone dictinitNone ver

boseFalse splitsignFalse ran

domstateNone positivecodeFalse

positivedictFalse

Dictionary learning

Finds a dictionary a set of atoms that can best be used to represent data using a sparse code

Solves the optimization problem

UV argmin 0.5 ||Y - UV||_F^2 alpha U 1

UV

with V_k 2 ||V_k||_F = 1 forall 0 < k < ncomponents

1622 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

ncomponents int number of dictionary elements to extract

alpha float sparsity controlling parameter

maxiter int maximum number of iterations to perform

tolfloat tolerance for numerical error

fitalgorithm 'lars' 'cd' lars uses the least angle regression method to solve the lasso problem linearmodellarspath cd uses the coordinate descent method to compute the Lasso solution linearmodelLasso Lars will be faster if the estimated components are sparse

New in version 017 cdcoordinate descent method to improve speed transformalgorithm 'lassolars' 'lassocd' 'lars' 'omp' 'threshold' Algorithm used to transform the data lars uses the least angle regression method linearmodellarspath lassolars uses Lars to compute the Lasso solution lassocd uses the coordinate descent method to compute the Lasso solution linearmodelLasso lassolars will be faster if the estimated components are sparse omp uses orthogonal matching pursuit to estimate the sparse solution threshold squashes to zero all coefficients less than alpha from the projectiondictionary X

New in version 017 lassocd coordinate descent method to improve speed transformnnonzerocoefs int01nfeatures by default Number of nonzero coefficients to target in each column of the solution This is only used by algorithmmlars andalgorithmmomp and is overridden by alpha in the Or thogonal Matching Pursuit OMP case transformalpha float 1 by default If algorithmmlassolars or algorithmmlassocd alpha is the penalty applied to the L1 norm If algorithmmthreshold alpha is the absolute value of the threshold below which coefficients will be squashed to zero If algorithmmomp alpha is the tolerance parameter the value of the reconstruction error targeted In this case it overrides nnonzerocoefs

njobs int or None optional defaultNone Number of parallel jobs to run None means 1

unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

codeinit array of shape nsamples ncomponents initial value for the code for warm

restart

dictinit array of shape ncomponents nfeatures initial values for the dictionary for warm

restart

verbose bool optional default False To control the verbosity of the procedure

splitsign bool False by default Whether to split the sparse feature vector into the concate nation of its negative part and its positive part This can improve the performance of down stream classifiers

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

69sklearndecomposition Matrix Decomposition 1623

scikitlearn user guide Release 0213

positivecode bool Whether to enforce positivity when finding the code
New in version 020

positivedict bool Whether to enforce positivity when finding the dictionary
New in version 020

Attributes

components array ncomponents nfeatures dictionary atoms extracted from the data

error array vector of errors at each iteration

niter int Number of iterations run

See also

SparseCoder

MiniBatchDictionaryLearning

SparsePCA

MiniBatchSparsePCA

Notes

References

J Mairal F Bach J Ponce G Sapiro 2009 Online dictionary learning for sparse coding <https://www.diens.fr/sierrapdf/sicml09pdf>

Methods

fitself X y Fit the model from data in X

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X Encode the data as a sparse combination of the dictionary atoms

init self ncomponents=None alpha=1 maxiter=1000 tol=1e-08 fit_algorithm='lars'

transform_algorithm='omp' transform_min_nonzero_coefs=None transform_alpha=None

n_jobs=None code_init=None dict_init=None verbose=False split_sign=False random_state=None positive_code=False positive_dict=False

fitself X y None

Fit the model from data in X

Parameters

X array-like shape (n_samples, n_features) Training vector where n_samples is the number of samples and n_features is the number of features

y ignored

Returns

self object Returns the object itself

1624 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Encode the data as a sparse combination of the dictionary atoms

Coding method is determined by the object parameter transformalgorithm

Parameters

Xarray of shape nsamples nfeatures Test data to be transformed must have the same number of features as the data used to train the model

Returns

Xnew array shape nsamples ncomponents Transformed data

692sklearndecomposition FactorAnalysis

classsklearndecomposition FactorAnalysis ncomponentsNone tol001 copyTrue

maxiter1000 noisevarianceinitNone

svdmethod'randomized' iteratedpower3

randomstate0

Factor Analysis FA

A simple linear generative model with Gaussian latent variables

The observations are assumed to be caused by a linear transformation of lower dimensional latent factors and added Gaussian noise Without loss of generality the factors are distributed according to a Gaussian with zero mean and unit covariance The noise is also zero mean and has an arbitrary diagonal covariance matrix

69sklearndecomposition Matrix Decomposition 1625

scikitlearn user guide Release 0213

If we would restrict the model further by assuming that the Gaussian noise is even isotropic all diagonal entries are the same we would obtain PPCA

FactorAnalysis performs a maximum likelihood estimate of the so called loading matrix the transformation of the latent variables to the observed ones using expectationmaximization EM

Read more in the User Guide

Parameters

ncomponents int None Dimensionality of latent space the number of components of Xthat are obtained after transform If None ncomponents is set to the number of features

tolfloat Stopping tolerance for EM algorithm

copy bool Whether to make a copy of X If False the input X gets overwritten during fitting

maxiter int Maximum number of iterations

noisevarianceinit None array shapenfeatures The initial guess of the noise variance for each feature If None it defaults to np.onesnfeatures

svdmethod 'lapack' 'randomized' Which SVD method to use If 'lapack' use standard SVD from scipy.linalg if 'randomized' use fast randomizedsvd function Defaults to 'randomized' For most applications 'randomized' will be sufficiently precise while providing significant speed gains Accuracy can also be improved by setting higher values for iteratedpower If this is not sufficient for maximum precision you should choose 'lapack'

iteratedpower int optional Number of iterations for the power method 3 by default Only used ifsvdmethod equals 'randomized'

randomstate int RandomState instance or None optional default0 If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random Only used when svdmethod equals 'randomized'

Attributes

components array ncomponents nfeatures Components with maximum variance

loglike list niterations The log likelihood at each iteration

noisevariance array shapenfeatures The estimated noise variance for each feature

niter int Number of iterations run

See also

PCA (Principal Component Analysis) Principal component analysis is also a latent linear variable model which however assumes equal noise variance for each feature This extra assumption makes probabilistic PCA (Principal Component Analysis) faster as it can be computed in closed form

FastICA Independent component analysis a latent variable model with nonGaussian latent variables

References

Examples

1626 Chapter 6 API (Application Programming Interface) Reference


```
scikitlearn user guide Release 0213
from sklearndatasets import loaddigits
from sklearndecomposition import FactorAnalysis
X = loaddigits(returnXy=True)
transformer = FactorAnalysis(n_components=7, random_state=0)
X_transformed = transformer.fit_transform(X)
X_transformed.shape
1797 7
Methods
fit(self, X, y) Fit the FactorAnalysis model to X using EM
fit_transform(self, X, y) Fit to data then transform it
get_covariance(self) Compute data covariance with the FactorAnalysis model
get_params(self, deep=True) Get parameters for this estimator
get_precision(self) Compute data precision matrix with the FactorAnalysis model
score(self, X, y) Compute the average loglikelihood of the samples
score_samples(self, X) Compute the loglikelihood of each sample
set_params(self, params) Set the parameters of this estimator
transform(self, X) Apply dimensionality reduction to X using the model
init(self, n_components=None, tol=0.01, copy=True, max_iter=1000, noise_variance=init_noise_variance, svd_method='randomized', iterated_power=3, random_state=0)
fit(self, X, y=None) Fit the FactorAnalysis model to X using EM
Parameters
X array-like of shape (n_samples, n_features) Training data
y ignored
Returns
self
fit_transform(self, X, y=None, fit_params=None) Fit to data then transform it
fit_transformer(self, X, y=None, fit_params=None) Fits transformer to X and y with optional parameters fit_params and returns a transformed version of X
Parameters
X numpy array of shape (n_samples, n_features) Training set
y numpy array of shape (n_samples,) Target values
Returns
X_new numpy array of shape (n_samples, n_features_new) Transformed array
get_covariance(self) Compute data covariance with the FactorAnalysis model
cov, components_T, components, diagonal_noise_variance
69sklearndecomposition Matrix Decomposition 1627
```

scikitlearn user guide Release 0213

Returns

cov array shape nfeatures nfeatures Estimated covariance of data

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Compute data precision matrix with the FactorAnalysis model

Returns

precision array shape nfeatures nfeatures Estimated precision of data

scoresselfXyNone

Compute the average loglikelihood of the samples

Parameters

Xarray shape nsamples nfeatures The data

yIgnored

Returns

lfloat Average loglikelihood of the samples under the current model

scoresamples selfX

Compute the loglikelihood of each sample

Parameters

Xarray shape nsamples nfeatures The data

Returns

larray shape nsamples Loglikelihood of each sample under the current model

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Apply dimensionality reduction to X using the model

Compute the expected mean of the latent variables See Barber 21233 or Bishop 1266

Parameters

Xarraylike shape nsamples nfeatures Training data

Returns

1628 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xnew arraylike shape nsamples ncomponents The latent variables of X

Examples using sklearn.decomposition.FactorAnalysis

- Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA

- Faces dataset decompositions

693sklearn.decomposition.FastICA

class sklearn.decomposition.FastICA ncomponents=None algorithm='parallel' whiten=True

fun='logcosh' funargs=None maxiter=200 tol=0.0001

winit=None randomstate=None

FastICA a fast algorithm for Independent Component Analysis

Read more in the User Guide

Parameters

ncomponents int optional Number of components to use If none is passed all are used

algorithm 'parallel' 'deflation' Apply parallel or deflational algorithm for FastICA

whiten boolean optional If whiten is false the data is already considered to be whitened and no whitening is performed

fun string or function optional Default 'logcosh' The functional form of the G function used in the approximation to negentropy Could be either 'logcosh' 'exp' or 'cube' You can also provide your own function It should return a tuple containing the value of the function

and of its derivative in the point Example

```
def mygx return x3 - 3 x2 meanaxis=1
```

funargs dictionary optional Arguments to send to the functional form If empty and if

fun='logcosh' funargs will take value 'alpha' 10

maxiter int optional Maximum number of iterations during fit

tol float optional Tolerance on update at each iteration

winit None of an ncomponents ncomponents ndarray The mixing matrix to be used to initialize the algorithm

randomstate int RandomState instance or None optional default=None If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random

Attributes

components 2D array shape ncomponents nfeatures The unmixing matrix

mixing array shape nfeatures ncomponents The mixing matrix

niter int If the algorithm is "deflation" niter is the maximum number of iterations run across all components Else they are just the number of iterations taken to converge

69sklearn.decomposition.MatrixDecomposition 1629

scikitlearn user guide Release 0213

Notes

Implementation based on A Hyvarinen and E Oja Independent Component Analysis Algorithms and Applications Neural Networks 1345 2000 pp 411430

Examples

```
from sklearn.datasets import load_digits
from sklearn.decomposition import FastICA
X, load_digits_return_Xy = True
transformer = FastICA(n_components=7,
                      random_state=0)
X_transformed = transformer.fit_transform(X)
X_transformed.shape
1797 7
```

Methods

```
fit(self, X, y) Fit the model to X
fit_transform(self, X, y) Fit the model and recover the sources from X
get_params(self, deep=True) Get parameters for this estimator
inverse_transform(self, X) Transform the sources back to the mixed data applying
mixing matrix
set_params(self, **params) Set the parameters of this estimator
transform(self, X) Recover the sources from X applying the unmixing matrix
init(self, n_components=None, algorithm='parallel', whiten=True, fun='logcosh',
funargs=None, max_iter=200, tol=0.00001, winit=None, random_state=None)
fit(self, X, y=None)
Fit the model to X
```

Parameters

X: array-like shape (n_samples, n_features) Training data where n_samples is the number of samples and n_features is the number of features
y: ignored
Returns
self

fit_transform(self, X, y=None)
Fit the model and recover the sources from X

Parameters

X: array-like shape (n_samples, n_features) Training data where n_samples is the number of samples and n_features is the number of features
y: ignored
Returns

scikitlearn user guide Release 0213

Xnew arraylike shape nsamples ncomponents

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfXcopyTrue

Transform the sources back to the mixed data apply mixing matrix

Parameters

Xarraylike shape nsamples ncomponents Sources where nsamples is the number

of samples and ncomponents is the number of components

copy bool optional If False data passed to fit are overwritten Defaults to True

Returns

Xnew arraylike shape nsamples nfeatures

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfXcopyTrue

Recover the sources from X apply the unmixing matrix

Parameters

Xarraylike shape nsamples nfeatures Data to transform where nsamples is the

number of samples and nfeatures is the number of features

copy bool optional If False data passed to fit are overwritten Defaults to True

Returns

Xnew arraylike shape nsamples ncomponents

Examples using sklearndecompositionFastICA

•Blind source separation using FastICA

•FastICA on 2D point clouds

•Faces dataset decompositions

69sklearndecomposition Matrix Decomposition 1631

scikitlearn user guide Release 0213

694sklearn.decomposition.IncrementalPCA

class sklearn.decomposition.IncrementalPCA ncomponents=None whiten=False copy=True
batchsize=None

Incremental principal components analysis IPCA

Linear dimensionality reduction using Singular Value Decomposition of the data keeping only the most significant singular vectors to project the data to a lower dimensional space The input data is centered but not scaled for each feature before applying the SVD

Depending on the size of the input data this algorithm can be much more memory efficient than a PCA (Principal Component Analysis)

This algorithm has constant memory complexity on the order of batchsize enabling use of mmap files without loading the entire file into memory

The computational overhead of each SVD is $O(\text{batchsize} \times \text{nfeatures}^2)$ but only $2 \times \text{batchsize} \times \text{nfeatures}$ samples remain in memory at a time There will be $\text{nfeatures} \times \text{batchsize}$ SVD computations to get the principal components versus 1 large SVD of complexity $O(\text{nfeatures}^2)$ for PCA (Principal Component Analysis)

Read more in the User Guide

Parameters

ncomponents int or None default=None Number of components to keep If ncomponents is None then ncomponents is set to min(nsamples, nfeatures)

whiten bool optional When True False by default the components vectors are divided by $\sqrt{\text{nsamples} \times \text{variance}}$ to ensure uncorrelated outputs with unit componentwise variances

Whitening will remove some information from the transformed signal the relative variance scales of the components but can sometimes improve the predictive accuracy of the downstream estimators by making data respect some hardwired assumptions

copy bool default=True If False X will be overwritten copy=False can be used to save memory but is unsafe for general use

batchsize int or None default=None The number of samples to use for each batch Only used when calling fit If batchsize is None then batchsize is inferred from the data and set to $\sqrt{\text{nfeatures}}$ to provide a balance between approximation accuracy and memory consumption

Attributes

components array shape (ncomponents, nfeatures) Components with maximum variance explained
variance array shape (ncomponents,) Variance explained by each of the selected components

explained_variance_ratio array shape (ncomponents,) Percentage of variance explained by each of the selected components If all components are stored the sum of explained variances is equal to 1.0

singular_values array shape (ncomponents,) The singular values corresponding to each of the selected components The singular values are equal to the $\sqrt{2}$ norms of the ncomponents variables in the lower dimensional space

mean array shape (nfeatures,) Per feature empirical mean aggregate over calls to partial_fit

1632 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

var array shape nfeatures Perfeature empirical variance aggregate over calls to partialfit

noisevariance float The estimated noise covariance following the Probabilistic PCA (Principal Component Analysis) model from Tipping and Bishop 1999 See “Pattern Recognition and Machine Learning” by C Bishop 1221 p 574 or <http://www.miketipping.com/papers/metmppcapdf>

ncomponents int The estimated number of components Relevant when ncomponents=None

nsamplesseen int The number of samples processed by the estimator Will be reset on new calls to fit but increments across partialfit calls

See also

PCA (Principal Component Analysis)

KernelPCA

SparsePCA

TruncatedSVD

Notes

Implements the incremental PCA (Principal Component Analysis) model from D Ross J Lim R Lin M Yang Incremental Learning for Robust Visual Tracking International Journal of Computer Vision Volume 77 Issue 13 pp 125141 May 2008 See <https://www.cstor.utoronto.edu/drossi/vt/RossLimLinYangijcv.pdf>

This model is an extension of the Sequential KarhunenLoeve Transform from A Levy and M Lindenbaum Sequential KarhunenLoeve Basis Extraction and its Application to Images IEEE Transactions on Image Processing Volume 9 Number 8 pp 13711374 August 2000 See <https://www.cstechnion.ac.il/mic/docs/kl.pdf>

We have specifically abstained from an optimization used by authors of both papers a QR decomposition used in specific situations to reduce the algorithmic complexity of the SVD The source for this technique is Matrix Computations Third Edition G Golub and C Van Loan Chapter 5 section 544 pp 252253 This technique has been omitted because it is advantageous only when decomposing a matrix with nsamples rows 53 nfeatures columns and hurts the readability of the implemented algorithm This would be a good opportunity for future optimization if it is deemed necessary

References

D Ross J Lim R Lin M Yang Incremental Learning for Robust Visual Tracking International Journal of Computer Vision Volume 77 Issue 13 pp 125141 May 2008

G Golub and C Van Loan Matrix Computations Third Edition Chapter 5 Section 544 pp 252253

Examples

```
from sklearn.datasets import load_digits
from sklearn.decomposition import IncrementalPCA
X, load_digits_return_Xy = load_digits(return_Xy=True)
transformer = IncrementalPCAncomponents=7 batchsize=200
# either partially fit on smaller batches of data
transformer.partial_fit(X[100:110])
IncrementalPCAbatchsize=200 copy=True ncomponents=7 whiten=False
69sklearn.decomposition.MatrixDecomposition 1633
```

scikitlearn user guide Release 0213

or let the fit function itself divide the data into batches

Xtransformed transformerfittransformX

Xtransformedshape

1797 7

Methods

fitself X y Fit the model with X using minibatches of size batchsize

fittransform self X y Fit to data then transform it

getcovariance self Compute data covariance with the generative model

getparams self deep Get parameters for this estimator

getprecision self Compute data precision matrix with the generative model

inversetransform self X Transform data back to its original space

partialfit self X y checkinput Incremental fit with X

setparams self params Set the parameters of this estimator

transform self X Apply dimensionality reduction to X

init selfncomponentsNone whitenFalse copyTrue batchsizeNone

fitselfXyNone

Fit the model with X using minibatches of size batchsize

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yIgnored

Returns

self object Returns the instance itself

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getcovariance self

Compute data covariance with the generative model

cov componentsT S2components sigma2 eyenfeatures

where S2 contains the explained variances and sigma2 contains the noise variances

Returns

cov array shapenfeatures nfeatures Estimated covariance of data

1634 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getparams selfdeepTrue
Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self
Compute data precision matrix with the generative model
Equals the inverse of the covariance but computed with the matrix inversion lemma for efficiency

Returns

precision array shapenfeatures nfeatures Estimated precision of data

inversetransform selfX
Transform data back to its original space
In other words return an input Xoriginal whose transform would be X

Parameters

Xarraylike shape nsamples ncomponents New data where nsamples is the number of samples and ncomponents is the number of components

Returns

Xoriginal arraylike shape nsamples nfeatures

Notes

If whitening is enabled inversetransform will compute the exact inverse operation which includes reversing whitening

partialfit selfXyNone checkinputTrue
Incremental fit with X All of X is processed as a single batch

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

checkinput bool Run checkarray on X

ylgnored

Returns

self object Returns the instance itself

setparams selfparams
Set the parameters of this estimator
The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

69sklearn decomposition Matrix Decomposition 1635

self
transform selfX
Apply dimensionality reduction to X
X is projected on the first principal components previously extracted from a training set
Parameters
Xarraylike shape nsamples nfeatures New data where nsamples is the number of samples and nfeatures is the number of features
Returns
Xnew arraylike shape nsamples ncomponents
Examples
import numpy as np
from sklearn.decomposition import IncrementalPCA
X = np.array([1, 2, 1, 3, 2, 1, 1, 2, 1, 3, 2])
ipca = IncrementalPCA(ncomponents=2, batchsize=3)
ipca.fit(X)
IncrementalPCA(batchsize=3, copy=True, ncomponents=2, whiten=False)
ipca.transform(X)
Examples using sklearn.decomposition.IncrementalPCA
•Incremental PCA (Principal Component Analysis)
695sklearn.decomposition.KernelPCA
class sklearn.decomposition.KernelPCA(ncomponents=None, kernel='linear', gamma=None, degree=3, coef0=1, kernel_params=None, alpha=10, fit_inverse_transform=False, eigensolver='auto', tol=0, max_iter=None, remove_zero_eig=False, random_state=None, copy_X=True, n_jobs=None)
Kernel Principal component analysis KPCA
Nonlinear dimensionality reduction through the use of kernels see Pairwise metrics Affinities and Kernels
Read more in the User Guide
Parameters
ncomponents int default=None Number of components If None all nonzero components are kept
kernel "linear" "poly" "rbf" "sigmoid" "cosine" "precomputed" Kernel De fault "linear"
gamma float default=1 Kernel coefficient for rbf poly and sigmoid kernels Ig nored by other kernels
degree int default=3 Degree for poly kernels Ignored by other kernels
coef0 float default=1 Independent term in poly and sigmoid kernels Ignored by other kernels
1636 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

kernelparams mapping of string to any defaultNone Parameters keyword arguments and values for kernel passed as callable object Ignored by other kernels

alpha int default10 Hyperparameter of the ridge regression that learns the inverse transform when fitinversetransformTrue

fitinversetransform bool defaultFalse Learn the inverse transform for nonprecomputed kernels ie learn to find the preimage of a point

eigensolver string ‘auto’‘dense’‘arpack’ default‘auto’ Select eigensolver to use If ncomponents is much less than the number of training samples arpack may be more efficient than the dense eigensolver

tolfloat default0 Convergence tolerance for arpack If 0 optimal value will be chosen by arpack

maxiter int defaultNone Maximum number of iterations for arpack If None optimal value will be chosen by arpack

removezeroeig boolean defaultFalse If True then all components with zero eigenvalues are removed so that the number of components in the output may be ncomponents and sometimes even zero due to numerical instability When ncomponents is None this parameter is ignored and components with zero eigenvalues are removed regardless

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom Used when eigensolver ‘arpack’

New in version 018

copyX boolean defaultTrue If True input X is copied and stored by the model in the Xfit attribute If no further changes will be done to X setting copyXFalse saves memory by storing a reference

New in version 018

njobs int or None optional defaultNone The number of parallel jobs to run None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

New in version 018

Attributes

lambdas array ncomponents Eigenvalues of the centered kernel matrix in decreasing order If ncomponents andremovezeroeig are not set then all values are stored

alphas array nsamples ncomponents Eigenvectors of the centered kernel matrix If ncomponents andremovezeroeig are not set then all components are stored

dualcoef array nsamples nfeatures Inverse transform matrix Only available when fitinversetransform is True

Xtransformedfit array nsamples ncomponents Projection of the fitted data on the kernel principal components Only available when fitinversetransform is True

Xfit nsamples nfeatures The data used to fit the model If copyXFalse then Xfit is a reference This attribute is used for the calls to transform

69sklearn.decomposition Matrix Decomposition 1637

scikitlearn user guide Release 0213

References

Kernel PCA (Principal Component Analysis) was introduced in Bernhard Schoelkopf Alexander J Smola and KlausRobert Mueller 1999
Kernel principal component analysis In Advances in kernel methods MIT Press Cambridge MA USA

327352

Examples

```
from sklearndatasets import loaddigits
from sklearndecomposition import KernelPCA
X, loaddigitsreturnXy = True
transformer = KernelPCAncomponents=7 kernel=linear
Xtransformed = transformer.fittransform(X)
Xtransformed.shape
1797 7
```

Methods

```
fitself X y Fit the model from data in X
fittransform self X y Fit the model from data in X and transform X
getparams self deep Get parameters for this estimator
inversetransform self X Transform X back to original space
setparams self params Set the parameters of this estimator
transform self X Transform X
init selfncomponents=None kernel='linear' gamma=None degree=3 coef0=1 kernel_params=None alpha=10 fit_inversetransform=False eigensolver='auto' tol=0 max_iter=None remove_zero_eig=False random_state=None copy_X=True n_jobs=None
fitselfXy=None
Fit the model from data in X
Parameters
Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features
Returns
self object Returns the instance itself
fittransform selfXy=None params
Fit the model from data in X and transform X
Parameters
Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features
Returns
Xnew arraylike shape nsamples ncomponents
getparams selfdeep=True
Get parameters for this estimator
1638 Chapter 6 API (Application Programming Interface) Reference
```

scikitlearn user guide Release 0213

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfX

Transform X back to original space

Parameters

Xarraylike shape nsamples ncomponents

Returns

Xnew arraylike shape nsamples nfeatures

References

“Learning to Find Prelimages” G Baklr et al 2004

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

transform selfX

Transform X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Xnew arraylike shape nsamples ncomponents

Examples using sklearndecompositionKernelPCA

•Kernel PCA (Principal Component Analysis)

69sklearndecomposition Matrix Decomposition 1639

scikitlearn user guide Release 0213
696sklearn.decomposition.LatentDirichletAllocation
class sklearn.decomposition.LatentDirichletAllocation ncomponents=10
doctopicprior=None
topicwordprior=None
learningmethod='batch'
learningdecay=0.7
learningoffset=100
maxiter=10 batchsize=128
evaluate_every=1 to
talsamples=10000000
perptol=0.1
meanchange_tol=0.001
maxdocupdateiter=100
njobs=None verbose=0 ran
domstate=None
Latent Dirichlet Allocation with online variational Bayes algorithm
New in version 0.17
Read more in the User Guide
Parameters
ncomponents: int optional default=10 Number of topics
doctopicprior: float optional default=None Prior of document topic distribution θ
If the value is None defaults to $1/ncomponents$ In 1 this is called α
topicwordprior: float optional default=None Prior of topic word distribution β If
the value is None defaults to $1/ncomponents$ In 1 this is called η
learningmethod: 'batch' 'online' default='batch' Method used to update component
Only used in fit method In general if the data size is large the online update will be
much faster than the batch update
Valid options
batch Batch variational Bayes method Use all training data in
each EM update
Old components will be overwritten in each iteration
online Online variational Bayes method In each EM update use
minibatch of training data to update the components
variable incrementally The learning rate is controlled by the
learningdecay and the learningoffset parameters
Changed in version 0.20 The default learning method is now batch
learningdecay: float optional default=0.7 It is a parameter that control learning rate in the
online learning method The value should be set between 0.5 1.0 to guarantee asymptotic
convergence When the value is 0.0 and batchsize is nsamples the update method is
same as batch learning In the literature this is called κ
learningoffset: float optional default=10 A positive parameter that downweights early
iterations in online learning It should be greater than 10 In the literature this is called
 τ_0
maxiter: integer optional default=10 The maximum number of iterations
batchsize: int optional default=128 Number of documents to use in each EM iteration
Only used in online learning
1640 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

`evaluateevery` int optional default0 How often to evaluate perplexity Only used in fit method set it to 0 or negative number to not evaluate perplexity in training at all Evaluating perplexity can help you check convergence in training process but it will also increase total training time Evaluating perplexity in every iteration might increase training time up to twofold

`totalsamples` int optional default1e6 Total number of documents Only used in the `partialfit` method

`perptol` float optional default1e1 Perplexity tolerance in batch learning Only used when `evaluateevery` is greater than 0

`meanchangetol` float optional default1e3 Stopping tolerance for updating document topic distribution in Estep

`maxdocupdateiter` int default100 Max number of iterations for updating document topic distribution in the Estep

`njobs` int or None optional defaultNone The number of jobs to use in the Estep None means 1 unless in a `joblibparallelbackend` context1means using all proces

sors See Glossary for more details

`verbose` int optional default0 Verbosity level

`randomstate` int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by `nprandom`

Attributes

`components` array ncomponents nfeatures Variational parameters for topic word dis

tribution Since the complete conditional for topic word distribution is a Dirichlet

`components[i, j]` can be viewed as pseudocount that represents the number of times

`word[j]` was assigned to topic `i` It can also be viewed as distribution over the words

for each topic after normalization `model.components_` `model.components`

`sumaxis1` `np.newaxis`

`nbatchiter` int Number of iterations of the EM step

`niter` int Number of passes over the dataset

References

1 “Online Learning for Latent Dirichlet Allocation” Matthew D Hoffman David M Blei Francis Bach 2010

2 “Stochastic Variational Inference” Matthew D Hoffman David M Blei Chong Wang John Paisley 2013

3 Matthew D Hoffman’s `onlinedavb` code Link <https://github.com/bleilab/onlinedavb>

Examples

```
from sklearn.decomposition import LatentDirichletAllocation
```

```
from sklearn.datasets import makemultilabelclassification
```

This produces a feature matrix of token counts similar to what

`CountVectorizer` would produce on text

69sklearn.decomposition Matrix Decomposition 1641

scikitlearn user guide Release 0213

X makemultilabelclassificationrandomstate0

lda LatentDirichletAllocationncomponents5

randomstate0

ldafitX

LatentDirichletAllocation

get topics for some given samples

ldatransformX2

array000360392 025499205 00036211 064236448 009541846

015297572 000362644 044412786 039568399 0003586

Methods

fitself X y Learn model for the data X with variational Bayes

method

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

partialfit self X y Online VB with MiniBatch update

perplexity self X subsampling Calculate approximate perplexity for data X

score self X y Calculate approximate loglikelihood as score

setparams self params Set the parameters of this estimator

transform self X Transform data X according to the fitted model

init selfncomponents10 doctopicpriorNone topicwordpriorNone learn

ingmethod'batch' learningdecay07 learningoffset100 maxiter10

batchsize128 evaluateevery1 totalsamples10000000 perptol01

meanchangeitol0001 maxdocupdateiter100 njobsNone verbose0 ran

domstateNone

fitselfXyNone

Learn model for the data X with variational Bayes method

Whenlearningmethod is 'online' use minibatch update Otherwise use batch update

Parameters

Xarraylike or sparse matrix shapensamples nfeatures Document word matrix

ylgnored

Returns

self

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

1642 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyNone

Online VB with MiniBatch update

Parameters

Xarraylike or sparse matrix shapensamples nfeatures Document word matrix

yIgnored

Returns

self

perplexity selfXsubsamplingFalse

Calculate approximate perplexity for data X

Perplexity is defined as $\exp(-\log\text{likelihood per word})$

Changed in version 0.19 doctopicdistr argument has been deprecated and is ignored because user no longer has access to unnormalized distribution

Parameters

Xarraylike or sparse matrix nsamples nfeatures Document word matrix

subsampling bool Do subsampling or not

Returns

score float Perplexity score

scoreselfXyNone

Calculate approximate loglikelihood as score

Parameters

Xarraylike or sparse matrix shapensamples nfeatures Document word matrix

yIgnored

Returns

score float Use approximate bound as score

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

69sklearn.decomposition Matrix Decomposition 1643

scikitlearn user guide Release 0213

transform selfX

Transform data X according to the fitted model

Changed in version 018 doctopicdistr is now normalized

Parameters

Xarraylike or sparse matrix shapensamples nfeatures Document word matrix

Returns

doctopicdistr shapensamples ncomponents Document topic distribution for X

Examples using sklearn.decomposition.LatentDirichletAllocation

- Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation

697sklearn.decomposition.MinibatchDictionaryLearning

class sklearn.decomposition.MinibatchDictionaryLearning(ncomponents=None, al

pha=1, niter=1000,

fit_algorithm='lars',

njobs=None,

batch_size=3, shuffle=True,

dict_init=None, trans

form_algorithm='omp', trans

form_nnonzero_coefs=None,

transform_alpha=None, ver

bose=False, split_sign=False,

random_state=None, pos

itive_code=False, posi

tive_dict=False)

Minibatch dictionary learning

Finds a dictionary a set of atoms that can best be used to represent data using a sparse code

Solves the optimization problem

$UV = \argmin_{U, V} \|Y - UV\|_2^2$ alpha U 1

UV

with $V_k \geq 0$ for all $0 \leq k < n_{components}$

Read more in the User Guide

Parameters

ncomponents int number of dictionary elements to extract

alpha float sparsity controlling parameter

niter int total number of iterations to perform

fit_algorithm {'lars', 'cd'} lars uses the least angle regression method to solve the lasso

problem linear_model.lars_path cd uses the coordinate descent method to compute the

Lasso solution linear_model.Lasso Lars will be faster if the estimated components are

sparse

1644 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

njobs int or None optional defaultNone Number of parallel jobs to run None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

batchsize int number of samples in each minibatch

shuffle bool whether to shuffle the samples before forming batches

dictinit array of shape ncomponents nfeatures initial value of the dictionary for warm restart scenarios

transformalgorithm 'lassolars' 'lassocd' 'lars' 'omp' 'threshold' Algorithm used to transform the data lars uses the least angle regression method linearmodellarspath lassolars uses Lars to compute the Lasso solution lassoed uses the coordinate descent method to compute the Lasso solution linearmodelLasso lassolars will be faster if the estimated components are sparse omp uses orthogonal matching pursuit to estimate the sparse solution threshold squashes to zero all coefficients less than alpha from the projection dictionary X'

transformnnonzerocoefs int01nfeatures by default Number of nonzero coefficients to target in each column of the solution This is only used by algorithmmlars andalgorithmmomp and is overridden by alpha in the Or thogonal Matching Pursuit OMP case

transformalpha float 1 by default If algorithmlassolars or algorithmlassocd alpha is the penalty applied to the L1 norm If algorithmthreshold alpha is the absolute value of the threshold below which coefficients will be squashed to zero If algorithmmomp alpha is the tolerance parameter the value of the reconstruction error targeted In this case it overrides nnonzerocoefs

verbose bool optional default False To control the verbosity of the procedure

splitsign bool False by default Whether to split the sparse feature vector into the concatenation of its negative part and its positive part This can improve the performance of downstream classifiers

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

positivecode bool Whether to enforce positivity when finding the code

New in version 020

positivedict bool Whether to enforce positivity when finding the dictionary

New in version 020

Attributes

components array ncomponents nfeatures components extracted from the data

innerstats tuple of A B ndarrays Internal sufficient statistics that are kept by the algorithm Keeping them is useful in online settings to avoid losing the history of the evolution but they shouldn't have any use for the end user A ncomponents ncomponents is the dictionary covariance matrix B nfeatures ncomponents is the data approximation matrix

niter int Number of iterations run

See also

69sklearnmatrixdecomposition Matrix Decomposition 1645

scikitlearn user guide Release 0213

SparseCoder

DictionaryLearning

SparsePCA

MiniBatchSparsePCA

Notes

References

J Mairal F Bach J Ponce G Sapiro 2009 Online dictionary learning for sparse coding <https://www.diens>

frsierrapdfsicml09pdf

Methods

fitself X y Fit the model from data in X

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

partialfit self X y iteroffset Updates the model using the data in X as a minibatch

setparams self params Set the parameters of this estimator

transform self X Encode the data as a sparse combination of the dictio

nary atoms

init self ncomponents None alpha 1 niter 1000 fit algorithm 'lars' njobs None

batchsize 3 shuffle True dict init None transform algorithm 'omp' trans

form nnonzero coeffs None transform alpha None verbose False splitsign False

randomstate None positivecode False positivedict False

fitself X y None

Fit the model from data in X

Parameters

X arraylike shape nsamples nfeatures Training vector where nsamples is the num

ber of samples and nfeatures is the number of features

y ignored

Returns

self object Returns the instance itself

fittransform self X y None fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

X numpy array of shape nsamples nfeatures Training set

y numpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

1646 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyNone iteroffsetNone

Updates the model using the data in X as a minibatch

Parameters

Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

yIgnored

iteroffset integer optional The number of iteration on data batches that has been performed before this call to partialfit This is optional if no number is passed the memory of the object is used

Returns

self object Returns the instance itself

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Encode the data as a sparse combination of the dictionary atoms

Coding method is determined by the object parameter transformalgorithm

Parameters

Xarray of shape nsamples nfeatures Test data to be transformed must have the same number of features as the data used to train the model

Returns

Xnew array shape nsamples ncomponents Transformed data

Examples using sklearndecompositionMiniBatchDictionaryLearning

- Image denoising using dictionary learning
- Faces dataset decompositions

69sklearndecomposition Matrix Decomposition 1647

scikitlearn user guide Release 0213

698sklearn.decomposition.MinibatchSparsePCA

class sklearn.decomposition.MinibatchSparsePCA ncomponents=None alpha=1

ridge_alpha=0.01 n_iter=100 call

back=None batch_size=3 verbose

shuffle=False n_jobs=None

method='lars' random_state=None

normalize_components=False

Minibatch Sparse Principal Components Analysis

Finds the set of sparse components that can optimally reconstruct the data. The amount of sparseness is controlled by the coefficient of the L1 penalty given by the parameter `alpha`.

Read more in the User Guide

Parameters

`ncomponents` int Number of sparse atoms to extract

`alpha` int Sparsity controlling parameter. Higher values lead to sparser components.

`ridge_alpha` float Amount of ridge shrinkage to apply in order to improve conditioning when calling the transform method.

`n_iter` int Number of iterations to perform for each mini batch.

`callback` callable or None optional default None callable that gets invoked every five iterations.

`batch_size` int the number of features to take in each mini batch.

`verbose` int Controls the verbosity: the higher, the more messages. Defaults to 0.

`shuffle` boolean whether to shuffle the data before splitting it in batches.

`n_jobs` int or None optional default None Number of parallel jobs to run. None means 1 unless in a `joblib.parallel_backend` context. 1 means using all processors. See Glossary for more details.

`method` 'lars' 'cd' 'lars' uses the least angle regression method to solve the lasso problem. 'linear_model.lars_path' 'cd' uses the coordinate descent method to compute the Lasso solution.

`linear_model.Lasso` 'lars' will be faster if the estimated components are sparse.

`random_state` int RandomState instance or None optional default None If int random

`state` is the seed used by the random number generator. If RandomState instance

`random_state` is the random number generator. If None the random number generator is the RandomState instance used by `np.random`.

`normalize_components` boolean optional default False

• if False use a version of Sparse PCA (Principal Component Analysis) without components normalization and without data centering. This is likely a bug and even though it's the default for backward compatibility this should not be used.

• if True use a version of Sparse PCA (Principal Component Analysis) with components normalization and data centering.

New in version 0.20

Deprecated since version 0.22 `normalize_components` was added and set to False

for backward compatibility. It would be set to True from 0.22 onwards.

Attributes

`components` array `ncomponents` `nfeatures` Sparse components extracted from the data.

1648 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

niter int Number of iterations run

mean array shape nfeatures Perfeature empirical mean estimated from the training set

Equal toXmeanaxis0

See also

PCA (Principal Component Analysis)

SparsePCA

DictionaryLearning

Examples

```
import numpy as np
from sklearn.datasets import makefriedman1
from sklearn.decomposition import MiniBatchSparsePCA
X = makefriedman1(nsamples=200, nfeatures=30, randomstate=0)
transformer = MiniBatchSparsePCA(ncomponents=5,
    batchsize=50,
    normalizecomponents=True,
    randomstate=0)
transformer.fit(X)
MiniBatchSparsePCA
Xtransformed = transformer.transform(X)
Xtransformed.shape
(200, 5)
```

most values in the components are zero sparsity

```
np.mean(transformer.components_, 0)
0.94
```

Methods

fitself X y Fit the model from data in X

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X Least Squares projection of the data onto the sparse components

```
init selfncomponents=None alpha=1 ridgealpha=0.01 niter=100 callback=None
batchsize=3 verbose=False shuffle=True njobs=None method='lars' ran
domstate=None normalizecomponents=False
fitselfXy=None
```

Fit the model from data in X

Parameters

Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

ylgnored

Returns

```
69sklearn.decomposition Matrix Decomposition 1649
```

scikitlearn user guide Release 0213

self object Returns the instance itself

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Least Squares projection of the data onto the sparse components

To avoid instability issues in case the system is underdetermined regularization can be applied Ridge

regression via the ridgealpha parameter

Note that Sparse PCA (Principal Component Analysis) components orthogonality is not enforced as in PCA (Principal Component Analysis) hence on

linear projection

Parameters

Xarray of shape nsamples nfeatures Test data to be transformed must have the same

number of features as the data used to train the model

Returns

Xnew array shape nsamples ncomponents Transformed data

Examples using sklearndecompositionMiniBatchSparsePCA

•Faces dataset decompositions

1650 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

699sklearnndecomposition NMF (Non-Negative Matrix Factorization)

classsklearnndecomposition NMFncomponentsNone initNone solver'cd'

betaloss'frobenius' tol00001 maxiter200 ran

domstateNone alpha00 l1ratio00 verbose0 shuf

fl=False

NonNegative Matrix Factorization NMF (Non-Negative Matrix Factorization)

Find two nonnegative matrices W H whose product approximates the non negative matrix X This factorization can be used for example for dimensionality reduction source separation or topic extraction

The objective function is

05X WHFro2

alpha l1ratio vecW1

alpha l1ratio vecH1

05alpha1 l1ratio WFro2

05alpha1 l1ratio HFro2

Where

AFro2 sumij Aij2 Frobenius norm

vecA1 sumij absAij Elementwise L1 norm

For multiplicativeupdate 'mu' solver the Frobenius norm 05 X WHFro2 can be changed into

another betadivergence loss by changing the betaloss parameter

The objective function is minimized with an alternating minimization of W and H

Read more in the User Guide

Parameters

ncomponents int or None Number of components if ncomponents is not set all features

are kept

init None 'random' 'nndsvd' 'nndsvda' 'nndsvdar' 'custom' Method used to initialize

the procedure Default None Valid options

•None 'nndsvd' if ncomponents minnsamples nfeatures otherwise random

•'random' nonnegative random matrices scaled with sqrtXmean

ncomponents

•'nndsvd' Nonnegative Double Singular Value Decomposition NNDSVD

initialization better for sparseness

•'nndsvda' NNDSVD with zeros filled with the average of X better when sparsity is

not desired

•'nndsvdar' NNDSVD with zeros filled with small random values generally faster

less accurate alternative to NNDSVDa for when sparsity is not desired

• 'custom' use custom matrices W and H

solver 'cd' 'mu' Numerical solver to use 'cd' is a Coordinate Descent solver 'mu' is a

Multiplicative Update solver

New in version 017 Coordinate Descent solver

New in version 019 Multiplicative Update solver

69sklearnndecomposition Matrix Decomposition 1651

scikitlearn user guide Release 0213

betaloss float or string default 'frobenius' String must be in 'frobenius' 'kullbackleibler' 'itakurasaito' Beta divergence to be minimized measuring the distance between X and the dot product WH Note that values different from 'frobenius' or 2 and 'kullbackleibler' or 1 lead to significantly slower fits Note that for betaloss 0 or 'itakurasaito' the input matrix X cannot contain zeros Used only in 'mu' solver

New in version 019

tolfloat default 1e4 Tolerance of the stopping condition

maxiter integer default 200 Maximum number of iterations before timing out

randomstate int RandomState instance or None optional default None If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

alpha double default 0 Constant that multiplies the regularization terms Set it to zero to

have no regularization

New in version 017 alpha used in the Coordinate Descent solver

l1ratio double default 0 The regularization mixing parameter with 0 l1ratio 1 For

l1ratio 0 the penalty is an elementwise L2 penalty aka Frobenius Norm For l1ratio

1 it is an elementwise L1 penalty For 0 l1ratio 1 the penalty is a combination of L1

and L2

New in version 017 Regularization parameter l1ratio used in the Coordinate Descent

solver

verbose bool defaultFalse Whether to be verbose

shuffle boolean default False If true randomize the order of coordinates in the CD solver

New in version 017 shuffle parameter used in the Coordinate Descent solver

Attributes

components array ncomponents nfeatures Factorization matrix sometimes called 'dic

tionary'

reconstructionerr number Frobenius norm of the matrix difference or betadivergence

between the training data Xand the reconstructed data WHfrom the fitted model

niter int Actual number of iterations

References

Cichocki Andrzej and P H A N AnhHuy "Fast local algorithms for large scale nonnegative matrix and ten

sor factorizations" IEICE transactions on fundamentals of electronics communications and computer sciences

923 708721 2009

Fevotte C Idier J 2011 Algorithms for nonnegative matrix factorization with the betadivergence Neural

Computation 239

Examples

1652 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

```
import numpy as np
```

```
X nparray1 1 2 1 3 12 4 1 5 08 6 1
```

```
from sklearn.decomposition import NMF (Non-Negative Matrix Factorization)
```

```
model NMFncomponents2 initrandom randomstate0
```

W modelfittransformX

H modelcomponents

Methods

fitself X y Learn a NMF (Non-Negative Matrix Factorization) model for the data X

fittransform self X y W H Learn a NMF (Non-Negative Matrix Factorization) model for the data X and returns the transformed data

```
getparams self deep Get parameters for this estimator
```

inversetransform self W Transform data back to its original space

setparams self params Set the parameters of this estimator

transform self X Transform the data X according to the fitted NMF (Non-Negative Matrix Factorization) model

```
init selfncomponentsNone initNone solver'cd' betaloss'frobenius' tol00001
```

```
maxiter200 randomstateNone alpha00 l1ratio00 verbose0 shuffleFalse
```

fitselfXyNone params

Learn a NMF (Non-Negative Matrix Factorization) model for the data X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Data matrix to be decom

posed

ylgnored

Returns

self

fittransform selfXyNone WNone HNone

Learn a NMF (Non-Negative Matrix Factorization) model for the data X and returns the transformed data

This is more efficient than calling `fit` followed by `transform`

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Data matrix to be decom

posed

ylgnored

Warraylike shape nsamples ncomponents If init='custom' it is used as initial guess for the solution

Arraylike shape ncomponents nfeatures If init='custom' it is used as initial guess for the solution

Returns

Warray shape nsamples ncomponents Transformed data

```
getparams selfdeepTrue
```

Get parameters for this estimator

Parameters

69sklearn.decomposition Matrix Decomposition 1653

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfW

Transform data back to its original space

Parameters

Warraylike sparse matrix shape nsamples ncomponents Transformed data matrix

Returns

Xarraylike sparse matrix shape nsamples nfeatures Data matrix of original shape

New in version 018

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Transform the data X according to the fitted NMF (Non-Negative Matrix Factorization) model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Data matrix to be trans

formed by the model

Returns

Warray shape nsamples ncomponents Transformed data

Examples using sklearn.decomposition.NMF

•Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation

•Selecting dimensionality reduction with Pipeline and GridSearchCV

•Faces dataset decompositions

6910sklearn.decomposition.PCA (Principal Component Analysis)

class sklearn.decomposition.PCA ncomponents=None copy=True whiten=False

svd_solver='auto' tol=0.0 iterated_power='auto' random_state=None

domstate=None

Principal component analysis PCA (Principal Component Analysis)

Linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space The input data is centered but not scaled for each feature before applying the SVD

It uses the LAPACK implementation of the full SVD or a randomized truncated SVD by the method of Halko

et al 2009 depending on the shape of the input data and the number of components to extract

1654 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

It can also use the scipysparselinalg ARPACK implementation of the truncated SVD

Notice that this class does not support sparse input See TruncatedSVD for an alternative with sparse data

Read more in the User Guide

Parameters

ncomponents int float None or string Number of components to keep if ncomponents is

not set all components are kept

ncomponents minnsamples nfeatures

Ifncomponents mle andsvdsolver full Minka’s MLE is used

to guess the dimension Use of ncomponents mle will interpret svdsolver

auto assvdsolver full

If0 ncomponents 1 andsvdsolver full select the number of

components such that the amount of variance that needs to be explained is greater than the

percentage specified by ncomponents

Ifsvdsolver arpack the number of components must be strictly less than the

minimum of nfeatures and nsamples

Hence the None case results in

ncomponents minnsamples nfeatures 1

copy bool default True If False data passed to fit are overwritten and running

fitXtransformX will not yield the expected results use fittransformX instead

whiten bool optional default False When True False by default the components vectors

are multiplied by the square root of nsamples and then divided by the singular values to

ensure uncorrelated outputs with unit componentwise variances

Whitening will remove some information from the transformed signal the relative variance

scales of the components but can sometime improve the predictive accuracy of the down

stream estimators by making their data respect some hardwired assumptions

svdsolver string ‘auto’ ‘full’ ‘arpack’ ‘randomized’

auto the solver is selected by a default policy based on Xshape andncomponents if

the input data is larger than 500x500 and the number of components to extract is lower

than 80 of the smallest dimension of the data then the more efficient ‘randomized’

method is enabled Otherwise the exact full SVD is computed and optionally truncated

afterwards

full run exact full SVD calling the standard LAPACK solver via scipylinalgsvd

and select the components by postprocessing

arpack run SVD truncated to ncomponents calling ARPACK solver via scipy

sparselinalgsvds It requires strictly 0 ncomponents minXshape

randomized run randomized SVD by the method of Halko et al

New in version 0180

tolfloat 0 optional default 0 Tolerance for singular values computed by svdsolver

‘arpack’

New in version 0180

69sklearnndecomposition Matrix Decomposition 1655

scikitlearn user guide Release 0213

iteratedpower int 0 or 'auto' default 'auto' Number of iterations for the power method computed by svdsolver 'randomized'

New in version 0180

randomstate int RandomState instance or None optional default None If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom Used when svdsolver 'arpack' or 'randomized'

New in version 0180

Attributes

components array shape ncomponents nfeatures Principal axes in feature space representing the directions of maximum variance in the data The components are sorted by explainedvariance

explainedvariance array shape ncomponents The amount of variance explained by each of the selected components

Equal to ncomponents largest eigenvalues of the covariance matrix of X

New in version 018

explainedvarianceratio array shape ncomponents Percentage of variance explained by each of the selected components

If ncomponents is not set then all components are stored and the sum of the ratios is equal to 10

singularvalues array shape ncomponents The singular values corresponding to each of the selected components The singular values are equal to the 2norms of the ncomponents variables in the lowerdimensional space

New in version 019

mean array shape nfeatures Perfeature empirical mean estimated from the training set Equal to Xmeanaxis0

ncomponents int The estimated number of components When ncomponents is set to 'mle' or a number between 0 and 1 with svdsolver 'full' this number is estimated from input data Otherwise it equals the parameter ncomponents or the lesser value of nfeatures and nsamples if ncomponents is None

noisevariance float The estimated noise covariance following the Probabilistic PCA (Principal Component Analysis) model from Tipping and Bishop 1999 See "Pattern Recognition and Machine Learning" by C Bishop 1221 p 574 or <http://www.miketipping.com/papers/metmppc.pdf> It is required to compute the estimated data covariance and score samples

Equal to the average of minnfeatures nsamples ncomponents smallest eigenvalues of the covariance matrix of X

See also

KernelPCA

SparsePCA

TruncatedSVD

IncrementalPCA

1656 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

References

For ncomponents 'mle' this class uses the method of Minka T P "Automatic choice of dimensionality for PCA (Principal Component Analysis)" In NIPS pp 598604
Implements the probabilistic PCA (Principal Component Analysis) model from Tipping M E and Bishop C M 1999 "Probabilistic principal component analysis" Journal of the Royal Statistical Society Series B Statistical Methodology 613 611622 via the score and scoresamples methods See <http://www.miketipping.com/papers/metmppca.pdf>
For svdsolver 'arpack' refer to scipysparselinalgsvds
For svdsolver 'randomized' see Halko N Martinsson P G and Tropp J A 2011 "Finding structure with randomness Probabilistic algorithms for constructing approximate matrix decompositions" SIAM review 532 217288 and also Martinsson P G Rokhlin V and Tygert M 2011 "A randomized algorithm for the decomposition of matrices" Applied and Computational Harmonic Analysis 301 4768

Examples

```
import numpy as np
from sklearn.decomposition import PCA (Principal Component Analysis)
X = np.array([1 2 1 3 2 1 1 2 1 3 2
pca = PCA(ncomponents=2
pca.fit(X
PCAcopy=True iteratedpower=auto ncomponents=2 randomstate=None
svdsolver=auto tol=0.0 whiten=False
print(pca.explained_variance_ratio_
0.9924 0.0075
print(pca.singular_values_
63.0061 0.54980
pca = PCA(ncomponents=2 svdsolver='full'
pca.fit(X
PCAcopy=True iteratedpower=auto ncomponents=2 randomstate=None
svdsolver='full' tol=0.0 whiten=False
print(pca.explained_variance_ratio_
0.9924 0.00755
print(pca.singular_values_
63.0061 0.54980
pca = PCA(ncomponents=1 svdsolver='arpack'
pca.fit(X
PCAcopy=True iteratedpower=auto ncomponents=1 randomstate=None
svdsolver='arpack' tol=0.0 whiten=False
print(pca.explained_variance_ratio_
0.99244
print(pca.singular_values_
63.0061
Methods
fit(self, X, y) Fit the model with X
Continued on next page
69sklearn.decomposition Matrix Decomposition 1657
```

fittransform self X y Fit the model with X and apply the dimensionality reduction on X

getcovariance self Compute data covariance with the generative model

getparams self deep Get parameters for this estimator

getprecision self Compute data precision matrix with the generative model

inversetransform self X Transform data back to its original space

score self X y Return the average loglikelihood of all samples

scoresamples self X Return the loglikelihood of each sample

setparams self params Set the parameters of this estimator

transform self X Apply dimensionality reduction to X

init self ncomponentsNone copyTrue whitenFalse svdsolver'auto' tol00 iter

atedpower'auto' randomstateNone

fitselfXyNone

Fit the model with X

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yIgnored

Returns

self object Returns the instance itself

fittransform selfXyNone

Fit the model with X and apply the dimensionality reduction on X

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yIgnored

Returns

Xnew arraylike shape nsamples ncomponents

getcovariance self

Compute data covariance with the generative model

cov componentsT S2components sigma2 eyenfeatures

where S2 contains the explained variances and sigma2 contains the noise variances

Returns

cov array shapenfeatures nfeatures Estimated covariance of data

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

scikitlearn user guide Release 0213

params mapping of string to any Parameter names mapped to their values

getprecision self

Compute data precision matrix with the generative model

Equals the inverse of the covariance but computed with the matrix inversion lemma for efficiency

Returns

precision array shapenfeatures nfeatures Estimated precision of data

inversetransform selfX

Transform data back to its original space

In other words return an input Xoriginal whose transform would be X

Parameters

Xarraylike shape nsamples ncomponents New data where nsamples is the number of samples and ncomponents is the number of components

Returns

Xoriginal arraylike shape nsamples nfeatures

Notes

If whitening is enabled inversetransform will compute the exact inverse operation which includes re

versing whitening

scoreselfXyNone

Return the average loglikelihood of all samples

See "Pattern Recognition and Machine Learning" by C Bishop 1221 p 574 or <http://www.miketipping.com/papers/metmppcapdf>

Parameters

Xarray shapensamples nfeatures The data

yIgnored

Returns

lfloat Average loglikelihood of the samples under the current model

scoresamples selfX

Return the loglikelihood of each sample

See "Pattern Recognition and Machine Learning" by C Bishop 1221 p 574 or <http://www.miketipping.com/papers/metmppcapdf>

Parameters

Xarray shapensamples nfeatures The data

Returns

larray shape nsamples Loglikelihood of each sample under the current model

setparams selfparams

Set the parameters of this estimator

69sklearn.decomposition Matrix Decomposition 1659

scikitlearn user guide Release 0213

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Apply dimensionality reduction to X

X is projected on the first principal components previously extracted from a training set

Parameters

Xarraylike shape nsamples nfeatures New data where nsamples is the number of samples and nfeatures is the number of features

Returns

Xnew arraylike shape nsamples ncomponents

Examples

```
import numpy as np
from sklearn.decomposition import IncrementalPCA
X = np.array([1, 2, 1, 3, 2, 1, 1, 2, 1, 3, 2])
ipca = IncrementalPCA(ncomponents=2, batchsize=3)
ipca.fit(X)
IncrementalPCA(batchsize=3, copy=True, ncomponents=2, whiten=False)
ipca.transform(X)
```

Examples using sklearn.decomposition.PCA

- Multilabel classification
- Explicit feature map approximation for RBF (Radial Basis Function) kernels
- Faces recognition example using eigenfaces and SVMs
- A demo of KMeans clustering on the handwritten digits data
- Concatenating multiple feature extraction methods
- Pipelining chaining a PCA (Principal Component Analysis) and a logistic regression
- Selecting dimensionality reduction with Pipeline and GridSearchCV
- The Iris Dataset
- PCA (Principal Component Analysis) example with Iris Dataset
- Incremental PCA (Principal Component Analysis)
- Comparison of LDA and PCA (Principal Component Analysis) 2D projection of Iris dataset
- Blind source separation using FastICA
- Principal components analysis PCA (Principal Component Analysis)
- FastICA on 2D point clouds
- Kernel PCA (Principal Component Analysis)

scikitlearn user guide Release 0213

- Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA
- Faces dataset decompositions
- Multidimensional scaling
- Balance model complexity and crossvalidated score
- Kernel Density Estimation
- Dimensionality Reduction with Neighborhood Components Analysis
- Using FunctionTransformer to select columns
- Importance of Feature Scaling

6911sklearn.decomposition.SparsePCA
class sklearn.decomposition.SparsePCA ncomponents=None alpha=1 ridge_alpha=0.01
max_iter=1000 tol=1e-08 method='lars'
n_jobs=None U=None V=None
verbose=False random_state=None normalize=False
fit_components=False

Sparse Principal Components Analysis SparsePCA

Finds the set of sparse components that can optimally reconstruct the data The amount of sparseness is controlled by the coefficient of the L1 penalty given by the parameter alpha

Read more in the User Guide

Parameters

ncomponents int Number of sparse atoms to extract
alpha float Sparsity controlling parameter Higher values lead to sparser components
ridge_alpha float Amount of ridge shrinkage to apply in order to improve conditioning when calling the transform method
max_iter int Maximum number of iterations to perform
tol float Tolerance for the stopping condition
method {'lars', 'cd'} lars uses the least angle regression method to solve the lasso problem linear_model.lars_path cd uses the coordinate descent method to compute the Lasso solution linear_model.Lasso Lars will be faster if the estimated components are sparse
n_jobs int or None optional default=None Number of parallel jobs to run None means 1 unless in a joblib.parallel_backend context 1 means using all processors See

Glossary for more details

U=None array of shape (n_samples, n_components) Initial values for the loadings for warm restart scenarios

V=None array of shape (n_components, n_features) Initial values for the components for warm restart scenarios

verbose int Controls the verbosity the higher the more messages Defaults to 0

random_state int RandomState instance or None optional default=None If int random

random_state is the seed used by the random number generator If RandomState instance

random_state is the random number generator If None the random number generator is the RandomState instance used by np.random

69sklearn.decomposition.MatrixDecomposition 1661

scikitlearn user guide Release 0213

normalizecomponents boolean optional defaultFalse

• if False use a version of Sparse PCA (Principal Component Analysis) without components normalization and without data centering This is likely a bug and even though it's the default for backward compatibility this should not be used

• if True use a version of Sparse PCA (Principal Component Analysis) with components normalization and data centering

New in version 020

Deprecated since version 022 normalizecomponents was added and set to False

for backward compatibility It would be set to True from 022 onwards

Attributes

components array ncomponents nfeatures Sparse components extracted from the data

error array Vector of errors at each iteration

niter int Number of iterations run

mean array shape nfeatures Perfeature empirical mean estimated from the training set

Equal toXmeanaxis0

See also

PCA (Principal Component Analysis)

MiniBatchSparsePCA

DictionaryLearning

Examples

import numpy as np

from sklearndatasets import makefriedman1

from sklearndecomposition import SparsePCA

X = makefriedman1nsamples200 nfeatures30 randomstate0

transformer = SparsePCAncomponents5

normalizecomponents True

randomstate0

transformerfitX

SparsePCA

Xtransformed = transformertransformX

Xtransformedshape

200 5

most values in the components are zero sparsity

npmeantransformercomponents 0

09666

Methods

fitself X y Fit the model from data in X

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

Continued on next page

1662 Chapter 6 API (Application Programming Interface) Reference

transform self X Least Squares projection of the data onto the sparse components

init selfncomponentsNone alpha1 ridgealpha001 maxiter1000 tol1e

08method'lars' njobsNone UinitNone VinitNone verboseFalse ran

domstateNone normalizecomponentsFalse

fitselfXyNone

Fit the model from data in X

Parameters

Xarraylike shape nsamples nfeatures Training vector where nsamples in the num

ber of samples and nfeatures is the number of features

yIgnored

Returns

self object Returns the instance itself

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Least Squares projection of the data onto the sparse components

To avoid instability issues in case the system is underdetermined regularization can be applied Ridge

regression via the ridgealpha parameter

69sklearndecomposition Matrix Decomposition 1663

scikitlearn user guide Release 0213

Note that Sparse PCA (Principal Component Analysis) components orthogonality is not enforced as in PCA (Principal Component Analysis) hence on linear projection

Parameters

Xarray of shape nsamples nfeatures Test data to be transformed must have the same number of features as the data used to train the model

Returns

Xnew array shape nsamples ncomponents Transformed data

6912sklearncomposition SparseCoder

classssklearncomposition SparseCoder dictionary transformalgorithm'omp'

transformnnonzerocoefsNone trans

formalphaNone splitsignFalse njobsNone

positivecodeFalse

Sparse coding

Finds a sparse representation of data against a fixed precomputed dictionary

Each row of the result is the solution to a sparse coding problem The goal is to find a sparse array code such that

X code dictionary

Read more in the User Guide

Parameters

dictionary array ncomponents nfeatures The dictionary atoms used for sparse coding

Lines are assumed to be normalized to unit norm

transformalgorithm 'lassolars' 'lassocd' 'lars' 'omp' 'threshold' Algorithm used to

transform the data lars uses the least angle regression method linearmodellarspath

lassolars uses Lars to compute the Lasso solution lassoed uses the coordinate descent

method to compute the Lasso solution linearmodelLasso lassolars will be faster if

the estimated components are sparse omp uses orthogonal matching pursuit to estimate

the sparse solution threshold squashes to zero all coefficients less than alpha from the

projectiondictionary X

transformnnonzerocoefs int01nfeatures by default Number of nonzero

coefficients to target in each column of the solution This is only used by

algorithmmlars andalgorithmmomp and is overridden by alpha in the Or

thogonal Matching Pursuit OMP case

transformalpha float 1 by default If algorithmlassolars or

algorithmlassocd alpha is the penalty applied to the L1 norm If

algorithmthreshold alpha is the absolute value of the threshold below

which coefficients will be squashed to zero If algorithmmomp alpha is the

tolerance parameter the value of the reconstruction error targeted In this case it overrides

nnonzerocoefs

splitsign bool False by default Whether to split the sparse feature vector into the concate

nation of its negative part and its positive part This can improve the performance of down

stream classifiers

1664 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

njobs int or None optional defaultNone Number of parallel jobs to run None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

positivecode bool Whether to enforce positivity when finding the code

New in version 020

Attributes

components array ncomponents nfeatures The unchanged dictionary atoms

See also

DictionaryLearning

MiniBatchDictionaryLearning

SparsePCA

MiniBatchSparsePCA

sparseencode

Methods

fitself X y Do nothing and return the estimator unchanged

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X Encode the data as a sparse combination of the dictionary atoms

init selfdictionary transformalgorithm'omp' transformmnnonzerocoefsNone transformalphaNone splitsignFalse njobsNone positivecodeFalse

fitselfXyNone

Do nothing and return the estimator unchanged

This method is just there to implement the usual API (Application Programming Interface) and hence work in pipelines

Parameters

XIgnored

yIgnored

Returns

self object Returns the object itself

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

69sklearnndecomposition Matrix Decomposition 1665

scikitlearn user guide Release 0213

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Encode the data as a sparse combination of the dictionary atoms

Coding method is determined by the object parameter transformalgorithm

Parameters

Xarray of shape nsamples nfeatures Test data to be transformed must have the same

number of features as the data used to train the model

Returns

Xnew array shape nsamples ncomponents Transformed data

Examples using sklearncompositionSparseCoder

•Sparse coding with a precomputed dictionary

6913sklearncomposition TruncatedSVD

classsklearncomposition TruncatedSVD ncomponents2 algorithm'randomized'

niter5 randomstateNone tol00

Dimensionality reduction using truncated SVD aka LSA

This transformer performs linear dimensionality reduction by means of truncated singular value decomposition

SVD Contrary to PCA (Principal Component Analysis) this estimator does not center the data before computing the singular value decompo

sition This means it can work with scipysparse matrices efficiently

In particular truncated SVD works on term counttfidf matrices as returned by the vectorizers in

sklearnfeatureextractiontext In that context it is known as latent semantic analysis LSA

This estimator supports two algorithms a fast randomized SVD solver and a "naive" algorithm that uses

ARPACK as an eigensolver on X XT or XT X whichever is more efficient

Read more in the User Guide

Parameters

1666 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

ncomponents int default 2 Desired dimensionality of output data Must be strictly less than the number of features The default value is useful for visualisation For LSA a value of 100 is recommended

algorithm string default "randomized" SVD solver to use Either "arpack" for the ARPACK wrapper in SciPy scipyparselinalgsvds or "randomized" for the randomized algorithm due to Halko 2009

niter int optional default 5 Number of iterations for randomized SVD solver Not used by ARPACK The default is larger than the default in randomizedsvd to handle sparse matrices that may have large slowly decaying spectrum

randomstate int RandomState instance or None optional default None If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random

tol float optional Tolerance for ARPACK 0 means machine precision Ignored by randomized SVD solver

Attributes

components array shape ncomponents nfeatures

explainedvariance array shape ncomponents The variance of the training samples transformed by a projection to each component

explainedvarianceratio array shape ncomponents Percentage of variance explained by each of the selected components

singularvalues array shape ncomponents The singular values corresponding to each of the selected components The singular values are equal to the 2-norms of the ncomponents variables in the lowerdimensional space

See also

PCA (Principal Component Analysis)

Notes

SVD suffers from a problem called "sign indeterminacy" which means the sign of the components and the output from transform depend on the algorithm and random state To work around this fit instances of this class to data once then keep the instance around to do transformations

References

Finding structure with randomness Stochastic algorithms for constructing approximate matrix decompositions

Halko et al 2009 arXiv909 <https://arxiv.org/pdf/0909.4061.pdf>

Examples

```
from sklearn.decomposition import TruncatedSVD
from sklearn.random_projection import sparse_random_matrix
X = sparse_random_matrix(100, 100, density=0.01, random_state=42)
svd = TruncatedSVD(n_components=5, n_iter=7, random_state=42)
svd.fit(X)
69sklearn.decomposition Matrix Decomposition 1667
```

scikitlearn user guide Release 0213
TruncatedSVDalgorithmrandomized ncomponents5 niter7
randomstate42 tol00
printsvdexplainedvarianceratio
00606 00584 00497 00434 00372
printsvdexplainedvarianceratiosum
0249
printsvdsingularvalues
25841 25245 23201 21753 20443
Methods
fitself X y Fit LSI model on training data X
fittransform self X y Fit LSI model to X and perform dimensionality reduc
tion on X
getparams self deep Get parameters for this estimator
inversetransform self X Transform X back to its original space
setparams self params Set the parameters of this estimator
transform self X Perform dimensionality reduction on X
init selfncomponents2 algorithm'randomized' niter5 randomstateNone tol00
fitselfXyNone
Fit LSI model on training data X
Parameters
Xarraylike sparse matrix shape nsamples nfeatures Training data
yIgnored
Returns
self object Returns the transformer object
fittransform selfXyNone
Fit LSI model to X and perform dimensionality reduction on X
Parameters
Xarraylike sparse matrix shape nsamples nfeatures Training data
yIgnored
Returns
Xnew array shape nsamples ncomponents Reduced version of X This will always
be a dense array
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values
1668 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

inversetransform selfX

Transform X back to its original space

Returns an array Xoriginal whose transform would be X

Parameters

Xarraylike shape nsamples ncomponents New data

Returns

Xoriginal array shape nsamples nfeatures Note that this is always a dense array

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Perform dimensionality reduction on X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data

Returns

Xnew array shape nsamples ncomponents Reduced version of X This will always

be a dense array

Examples using sklearndecompositionTruncatedSVD

- Column Transformer with Heterogeneous Data Sources

- Hashing feature transformation using Totally Random Trees

- Manifold learning on handwritten digits Locally Linear Embedding Isomap

- Clustering text documents using kmeans

decompositiondictlearning X

ncomponents Solves a dictionary learning matrix factorization problem

decompositiondictlearningonline X

Solves a dictionary learning matrix factorization problem

online

decompositionfastica X ncomponents Perform Fast Independent Component Analysis

decompositionnonnegativefactorization X Compute Nonnegative Matrix Factorization NMF (Non-Negative Matrix Factorization)

decompositionsparseencode X dictionary

Sparse coding

69sklearndecomposition Matrix Decomposition 1669

scikitlearn user guide Release 0213
6914sklearn.decomposition.DictionaryLearning
sklearn.decomposition.DictionaryLearning X: array-like, n_components: int, alpha: float, max_iter: int, tol: float, method: str, n_jobs: int, dict_init: array-like, code_init: array-like, callback: callable, verbose: int, random_state: int, return_niter: bool, positive_dict: bool, positive_code: bool
Solves a dictionary learning matrix factorization problem
Finds the best dictionary and the corresponding sparse code for approximating the data matrix X by solving
$$U, V = \arg\min_{U, V} \|X - UV\|_2^2$$

where V is the dictionary and U is the sparse code
Read more in the User Guide
Parameters
X: array-like of shape (n_samples, n_features) Data matrix
n_components: int Number of dictionary atoms to extract
alpha: float Sparsity controlling parameter
max_iter: int Maximum number of iterations to perform
tol: float Tolerance for the stopping condition
method: str {'lars', 'cd'} 'lars' uses the least angle regression method to solve the lasso problem 'cd' uses the coordinate descent method to compute the Lasso solution
linear_model: str {'lasso', 'lars', 'cd'} 'lasso' uses the least angle regression method to solve the lasso problem 'cd' uses the coordinate descent method to compute the Lasso solution
n_jobs: int or None optional default=None Number of parallel jobs to run None means 1 unless in a joblib.parallel_backend context 1 means using all processors See Glossary for more details
dict_init: array-like of shape (n_components, n_features) Initial value for the dictionary for warm restart scenarios
code_init: array-like of shape (n_samples, n_components) Initial value for the sparse code for warm restart scenarios
callback: callable or None optional default=None Callable that gets invoked every five iterations
verbose: bool optional default=False To control the verbosity of the procedure
random_state: int RandomState instance or None optional default=None If int random_state is the seed used by the random number generator If RandomState instance random_state is the random number generator If None the random number generator is the RandomState instance used by np.random
return_niter: bool Whether or not to return the number of iterations
positive_dict: bool Whether to enforce positivity when finding the dictionary
New in version 0.20
positive_code: bool Whether to enforce positivity when finding the code
New in version 0.20
1670 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns
code array of shape nsamples ncomponents The sparse code factor in the matrix factorization
dictionary array of shape ncomponents nfeatures The dictionary factor in the matrix factorization
errors array Vector of errors at each iteration
niter int Number of iterations run Returned only if returnniter is set to True

See also

dictlearningonline
DictionaryLearning
MiniBatchDictionaryLearning
SparsePCA
MiniBatchSparsePCA

6915sklearn.decomposition dictlearningonline
sklearn.decomposition dictlearningonline Xncomponents2 alpha1 niter100
returncodeTrue dictinitNone call
backNone batchsize3 verboseFalse
shuffleTrue njobsNone method'lars'
iteroffset0 randomstateNone re
turninnerstatsFalse innerstatsNone
returnniterFalse positivedictFalse
positivecodeFalse
Solves a dictionary learning matrix factorization problem online
Finds the best dictionary and the corresponding sparse code for approximating the data matrix X by solving
 $U V = \argmin_{U, V} \|X - UV\|_2^2$ alpha U 1
UV

with V_k 2 1 forall 0 k ncomponents
where V is the dictionary and U is the sparse code This is accomplished by repeatedly iterating over mini
batches by slicing the input data
Read more in the User Guide

Parameters
Xarray of shape nsamples nfeatures Data matrix
ncomponents int Number of dictionary atoms to extract
alpha float Sparsity controlling parameter
niter int Number of iterations to perform
returncode boolean Whether to also return the code U or just the dictionary V
dictinit array of shape ncomponents nfeatures Initial value for the dictionary for warm
restart scenarios
69sklearn.decomposition Matrix Decomposition 1671

scikitlearn user guide Release 0213

callback callable or None optional default None callable that gets invoked every five iterations

batchsize int The number of samples to take in each batch

verbose bool optional default False To control the verbosity of the procedure

shuffle boolean Whether to shuffle the data before splitting it in batches

njobs int or None optional defaultNone Number of parallel jobs to run None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

method 'lars' 'cd' lars uses the least angle regression method to solve the lasso problem linearmodellarspath cd uses the coordinate descent method to compute the Lasso solution linearmodelLasso Lars will be faster if the estimated components are sparse

iteroffset int default 0 Number of previous iterations completed on the dictionary used for initialization

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

returninnerstats boolean optional Return the inner statistics A dictionary covariance and B data approximation Useful to restart the algorithm in an online setting If returninnerstats is True returncode is ignored

innerstats tuple of A B ndarrays Inner sufficient statistics that are kept by the algorithm Passing them at initialization is useful in online settings to avoid losing the history of the evolution A ncomponents ncomponents is the dictionary covariance matrix B nfeatures ncomponents is the data approximation matrix

returnniter bool Whether or not to return the number of iterations

positivedict bool Whether to enforce positivity when finding the dictionary

New in version 020

positivecode bool Whether to enforce positivity when finding the code

New in version 020

Returns

code array of shape nsamples ncomponents the sparse code only returned if returncodeTrue

dictionary array of shape ncomponents nfeatures the solutions to the dictionary learning problem

niter int Number of iterations run Returned only if returnniter is set toTrue

See also

dictlearning

DictionaryLearning

MiniBatchDictionaryLearning

SparsePCA

MiniBatchSparsePCA

1672 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

6916sklearn.decomposition.fastica

sklearn.decomposition.fastica

Xcomponents=None algorithm='parallel' whiten=True

fun'logcosh' funargs=None maxiter=200 tol=0.0001

winit=None randomstate=None returnXmean=False

computesources=True returnniter=False

Perform Fast Independent Component Analysis

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

ncomponents int optional Number of components to extract If None no dimension reduction is performed

algorithm 'parallel' 'deflation' optional Apply a parallel or deflational FASTICA algorithm

whiten boolean optional If True perform an initial whitening of the data If False the data is assumed to have already been preprocessed it should be centered normed and white Otherwise you will get incorrect results In this case the parameter ncomponents will be ignored

fun string or function optional Default 'logcosh' The functional form of the G function used in the approximation to negentropy Could be either 'logcosh' 'exp' or 'cube' You can also provide your own function It should return a tuple containing the value of the function and of its derivative in the point The derivative should be averaged along its last dimension Example

```
def mygx return x 3 np.mean(x 2 axis=1)
```

funargs dictionary optional Arguments to send to the functional form If empty or None and if fun'logcosh' funargs will take value 'alpha' 10

maxiter int optional Maximum number of iterations to perform

tol float optional A positive scalar giving the tolerance at which the unmixing matrix is considered to have converged

winit ncomponents ncomponents array optional Initial unmixing array of dimension ncomponents

randomstate int RandomState instance or None optional default=None If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random

returnXmean bool optional If True Xmean is returned too

computesources bool optional If False sources are not computed but only the rotation matrix This can save memory when working with big data Defaults to True

returnniter bool optional Whether or not to return the number of iterations

Returns

Karray shape ncomponents nfeatures None If whiten is 'True' K is the prewhitening matrix that projects data onto the first ncomponents principal components If whiten is 'False' K is 'None'

69sklearn.decomposition Matrix Decomposition 1673

scikitlearn user guide Release 0213

Warray shape ncomponents ncomponents Estimated unmixing matrix The mixing matrix can be obtained by

w npdotW KT

A wT wwTl

Sarray shape nsamples ncomponents None Estimated source matrix

Xmean array shape nfeatures The mean over features Returned only if returnXmean is True

niter int If the algorithm is “deflation” niter is the maximum number of iterations run across all components Else they are just the number of iterations taken to converge This is returned only when returnniter is set to True

Notes

The data matrix X is considered to be a linear combination of nonGaussian independent components ie X

AS where columns of S contain the independent components and A is a linear mixing matrix In short ICA attempts to unmix the data by estimating an unmixing matrix W where S W

K X

This implementation was originally made for data of shape nfeatures nsamples Now the input is transposed before the algorithm is applied This makes it slightly faster for Fortranordered input

Implemented using FastICA A Hyvarinen and E Oja Independent Component Analysis Algorithms and

Applications Neural Networks 1345 2000 pp 411430

6917sklearndecomposition nonnegativefactorization

sklearndecomposition nonnegativefactorization X WNone HNone

ncomponentsNone init‘warn’

updateHTrue solver‘cd’

betaloss‘frobenius’ tol00001

maxiter200 alpha00

l1ratio00 regularizationNone

randomstateNone verbose0

shuffleFalse

Compute Nonnegative Matrix Factorization NMF (Non-Negative Matrix Factorization)

Find two nonnegative matrices W H whose product approximates the non negative matrix X This factorization can be used for example for dimensionality reduction source separation or topic extraction

The objective function is

05X WHFro2

alpha l1ratio vecW1

alpha l1ratio vecH1

05alpha1 l1ratio WFro2

05alpha1 l1ratio HFro2

Where

AFro2 sumij Aij2 Frobenius norm

vecA1 sumij absAij Elementwise L1 norm

1674 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

For multiplicativeupdate 'mu' solver the Frobenius norm $\|X - WH\|_F^2$ can be changed into another betadivergence loss by changing the betaloss parameter

The objective function is minimized with an alternating minimization of W and H If H is given and updateH=False it solves for W only

Parameters

Xarraylike shape nsamples nfeatures Constant matrix

Warraylike shape nsamples ncomponents If init='custom' it is used as initial guess for the solution

Harraylike shape ncomponents nfeatures If init='custom' it is used as initial guess for the solution If updateH=False it is used as a constant to solve for W only

ncomponents integer Number of components if ncomponents is not set all features are kept

init None 'random' 'nndsvd' 'nndsvda' 'nndsvdar' 'custom' Method used to initialize the procedure Default 'random'

The default value will change from 'random' to None in version 023 to make it consistent with decompositionNMF

Valid options

- None 'nndsvd' if ncomponents < nfeatures otherwise 'random'
- 'random' nonnegative random matrices scaled with sqrt(X.mean(ncomponents))

• 'nndsvd' Nonnegative Double Singular Value Decomposition NNDSVD

initialization better for sparseness

• 'nndsvda' NNDSVD with zeros filled with the average of X better when sparsity is not desired

• 'nndsvdar' NNDSVD with zeros filled with small random values generally faster

less accurate alternative to NNDSVDa for when sparsity is not desired

• 'custom' use custom matrices W and H

updateH boolean default True Set to True both W and H will be estimated from initial

guesses Set to False only W will be estimated

solver 'cd' 'mu' Numerical solver to use 'cd' is a Coordinate Descent solver that uses Fast Hierarchical

Alternating Least Squares Fast HALS

'mu' is a Multiplicative Update solver

New in version 017 Coordinate Descent solver

New in version 019 Multiplicative Update solver

betaloss float or string default 'frobenius' String must be in 'frobenius' 'kullbackleibler'

'itakurasaito' Beta divergence to be minimized measuring the distance between X and

the dot product WH Note that values different from 'frobenius' or 2 and 'kullbackleibler'

or 1 lead to significantly slower fits Note that for betaloss 0 or 'itakurasaito' the

input matrix X cannot contain zeros Used only in 'mu' solver

New in version 019

tolfloat default 1e-4 Tolerance of the stopping condition

69sklearn.decomposition Matrix Decomposition 1675

scikitlearn user guide Release 0213

maxiter integer default 200 Maximum number of iterations before timing out

alpha double default 0 Constant that multiplies the regularization terms

l1ratio double default 0 The regularization mixing parameter with 0 l1ratio = 1 For l1ratio = 0 the penalty is an elementwise L2 penalty aka Frobenius Norm For l1ratio = 1 it is an elementwise L1 penalty For 0 < l1ratio < 1 the penalty is a combination of L1 and L2

regularization 'both' 'components' 'transformation' None Select whether the regularization affects the components H the transformation W both or none of them

randomstate int RandomState instance or None optional default None If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random

verbose integer default 0 The verbosity level

shuffle boolean default False If true randomize the order of coordinates in the CD solver

Returns

Warraylike shape nsamples ncomponents Solution to the nonnegative least squares problem

Harraylike shape ncomponents nfeatures Solution to the nonnegative least squares problem

niter int Actual number of iterations

References

Cichocki Andrzej and P H A N AnhHuy "Fast local algorithms for large scale nonnegative matrix and tensor factorizations" IEICE transactions on fundamentals of electronics communications and computer sciences 923 708721 2009

Fevotte C Idier J 2011 Algorithms for nonnegative matrix factorization with the betadivergence Neural Computation 239

Examples

```
import numpy as np
```

```
X = np.array([1 2 1 3 12 4 1 5 08 6 1
```

```
from sklearn.decomposition import NonnegativeFactorization
```

```
W, H = NonnegativeFactorization(X, n_components=2
```

```
init_random, random_state=0
```

```
6918sklearn.decomposition.sparse_encode
```

```
sklearn.decomposition.sparse_encode(X, dictionary, gram=None, cov=None, algo
```

```
rithm='lasso', lars_n_nonzero_coefs=None, alpha
```

```
path=None, copy_cov=True, init=None, max_iter=1000
```

```
n_jobs=None, check_input=True, verbose=0, positive
```

```
definite=False
```

Sparse coding

1676 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Each row of the result is the solution to a sparse coding problem The goal is to find a sparse array code such that

X code dictionary

Read more in the User Guide

Parameters

Xarray of shape nsamples nfeatures Data matrix

dictionary array of shape ncomponents nfeatures The dictionary matrix against which to solve the sparse coding of the data Some of the algorithms assume normalized rows for meaningful output

gram array shapencomponents ncomponents Precomputed Gram matrix dictionary dictionary'

cov array shapencomponents nsamples Precomputed covariance dictionary' X

algorithm 'lassolars' 'lassocd' 'lars' 'omp' 'threshold' lars uses the least angle re

gression method linearmodellarspath lassolars uses Lars to compute the Lasso so

lution lassoed uses the coordinate descent method to compute the Lasso solution lin

earmodelLasso lassolars will be faster if the estimated components are sparse omp

uses orthogonal matching pursuit to estimate the sparse solution threshold squashes to zero

all coefficients less than alpha from the projection dictionary X'

nnonzerocefs int 01 nfeatures by default Number of nonzero coefficients to tar

get in each column of the solution This is only used by algorithmmlars and

algorithmmomp and is overridden by alpha in the Orthogonal Matching Pursuit

OMP case

alpha float 1 by default If algorithmmllassolars or

algorithmmllassocd alpha is the penalty applied to the L1 norm If

algorithmmthreshold alpha is the absolute value of the threshold below

which coefficients will be squashed to zero If algorithmmomp alpha is the

tolerance parameter the value of the reconstruction error targeted In this case it overrides

nnonzerocefs

copycov boolean optional Whether to copy the precomputed covariance matrix if False it may be overwritten

init array of shape nsamples ncomponents Initialization value of the sparse codes Only used ifalgorithmmllassocd

maxiter int 1000 by default Maximum number of iterations to perform if

algorithmmllassocd

njobs int or None optional defaultNone Number of parallel jobs to run None means 1

unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

checkinput boolean optional If False the input arrays X and dictionary will not be checked

verbose int optional Controls the verbosity the higher the more messages Defaults to 0

positive boolean optional Whether to enforce positivity when finding the encoding

New in version 020

Returns

code array of shape nsamples ncomponents The sparse codes

69sklearn decomposition Matrix Decomposition 1677

scikitlearn user guide Release 0213

See also

sklearnlinearmodellarspath

sklearnlinearmodelorthogonalmp

sklearnlinearmodelLasso

SparseCoder

610sklearnndiscriminantanalysis Discriminant Analysis

Linear Discriminant Analysis and Quadratic Discriminant Analysis

User guide See the Linear and Quadratic Discriminant Analysis section for further details

discriminantanalysis

LinearDiscriminantAnalysis Linear Discriminant Analysis

discriminantanalysis

QuadraticDiscriminantAnalysis Quadratic Discriminant Analysis

6101sklearnndiscriminantanalysis LinearDiscriminantAnalysis

classsklearnndiscriminantanalysis LinearDiscriminantAnalysis solver'svd'

shrinkageNone

priorsNone

ncomponentsNone

storecovarianceFalse

tol00001

Linear Discriminant Analysis

A classifier with a linear decision boundary generated by fitting class conditional densities to the data and using Bayes' rule

The model fits a Gaussian density to each class assuming that all classes share the same covariance matrix

The fitted model can also be used to reduce the dimensionality of the input by projecting it to the most discriminative directions

New in version 017 LinearDiscriminantAnalysis

Read more in the User Guide

Parameters

solver string optional

Solver to use possible values

- 'svd' Singular value decomposition default Does not compute the covariance matrix therefore this solver is recommended for data with a large number of features

- 'lsqr' Least squares solution can be combined with shrinkage

- 'eigen' Eigenvalue decomposition can be combined with shrinkage

shrinkage string or float optional

Shrinkage parameter possible values

1678 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- None no shrinkage default
- 'auto' automatic shrinkage using the LedoitWolf lemma
- float between 0 and 1 fixed shrinkage parameter

Note that shrinkage works only with 'lsqr' and 'eigen' solvers

priors array optional shape nclasses Class priors

ncomponents int optional defaultNone Number of components minnclasses

1 nfeatures for dimensionality reduction If None will be set to minnclasses 1

nfeatures

storecovariance bool optional Additionally compute class covariance matrix default

False used only in 'svd' solver

New in version 017

tolfloat optional default 10e4 Threshold used for rank estimation in SVD solver

New in version 017

Attributes

coef array shape nfeatures or nclasses nfeatures Weight vectors

intercept array shape nfeatures Intercept term

covariance arraylike shape nfeatures nfeatures Covariance matrix shared by all

classes

explainedvarianceratio array shape ncomponents Percentage of variance explained

by each of the selected components If ncomponents is not set then all components are

stored and the sum of explained variances is equal to 10 Only available when eigen or svd

solver is used

means arraylike shape nclasses nfeatures Class means

priors arraylike shape nclasses Class priors sum to 1

scalings arraylike shape rank nclasses 1 Scaling of the features in the space spanned

by the class centroids

xbar arraylike shape nfeatures Overall mean

classes arraylike shape nclasses Unique class labels

See also

sklearnndiscriminantanalysisQuadraticDiscriminantAnalysis Quadratic Discrimi

nant Analysis

Notes

The default solver is 'svd' It can perform both classification and transform and it does not rely on the calcu

lation of the covariance matrix This can be an advantage in situations where the number of features is large

However the 'svd' solver cannot be used with shrinkage

The 'lsqr' solver is an efficient algorithm that only works for classification It supports shrinkage

The 'eigen' solver is based on the optimization of the between class scatter to within class scatter ratio It can

be used for both classification and transform and it supports shrinkage However the 'eigen' solver needs to

compute the covariance matrix so it might not be suitable for situations with a high number of features

610sklearnndiscriminantanalysis Discriminant Analysis 1679

scikitlearn user guide Release 0213

Examples

```
import numpy as np
from sklearn.discriminantanalysis import LinearDiscriminantAnalysis
X = np.array([1, 2, 1, 3, 2, 1, 1, 2, 1, 3, 2])
y = np.array([1, 1, 1, 2, 2, 2])
clf = LinearDiscriminantAnalysis()
clf.fit(X, y)
LinearDiscriminantAnalysis(n_components=None, priors=None, shrinkage=None,
solver='svd', store_covariance=False, tol=0.0001)
print(clf.predict([0.8, 1.1]))
```

Methods

`decisionfunction` self X Predict confidence scores for samples
`fit` self X y Fit LinearDiscriminantAnalysis model according to the given training data and parameters
`fit_transform` self X y Fit to data then transform it
`get_params` self deep Get parameters for this estimator
`predict` self X Predict class labels for samples in X
`predict_log_proba` self X Estimate log probability
`predict_proba` self X Estimate probability
`score` self X y sample_weight Returns the mean accuracy on the given test data and labels
`set_params` self params Set the parameters of this estimator
`transform` self X Project data to maximize class separation
`__init__` self solver='svd' shrinkage=None priors=None n_components=None store_covariance=False tol=0.0001
`decisionfunction` self X
Predict confidence scores for samples
The confidence score for a sample is the signed distance of that sample to the hyperplane
Parameters
X array-like or sparse matrix shape (n_samples, n_features) Samples
Returns
array shape (n_samples, n_classes - 1) Confidence scores per sample class combination In the binary case confidence score for self.classes_[1] where 0 means this class would be predicted

`fit` self X y Fit LinearDiscriminantAnalysis model according to the given training data and parameters
Changed in version 0.19 store_covariance has been moved to main constructor
Changed in version 0.19 tol has been moved to main constructor

Parameters

X array-like shape (n_samples, n_features) Training data
1680 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

array shape nsamples Target values

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Predicted class label per sample

predictlogproba selfX

Estimate log probability

Parameters

Xarraylike shape nsamples nfeatures Input data

Returns

Carray shape nsamples nclasses Estimated log probabilities

predictproba selfX

Estimate probability

Parameters

Xarraylike shape nsamples nfeatures Input data

Returns

Carray shape nsamples nclasses Estimated probabilities

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

610sklearndiscriminantanalysis Discriminant Analysis 1681

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Project data to maximize class separation

Parameters

Xarraylike shape nsamples nfeatures Input data

Returns

Xnew array shape nsamples ncomponents Transformed data

Examples using sklearndiscriminantanalysisLinearDiscriminantAnalysis

- Normal and Shrinkage Linear Discriminant Analysis for classification
- Linear and Quadratic Discriminant Analysis with covariance ellipsoid
- Comparison of LDA and PCA (Principal Component Analysis) 2D projection of Iris dataset
- Manifold learning on handwritten digits Locally Linear Embedding Isomap
- Dimensionality Reduction with Neighborhood Components Analysis

6102sklearndiscriminantanalysis QuadraticDiscriminantAnalysis

classsklearndiscriminantanalysis QuadraticDiscriminantAnalysis priorsNone

regparam00

storecovarianceFalse

tol00001

Quadratic Discriminant Analysis

A classifier with a quadratic decision boundary generated by fitting class conditional densities to the data and using Bayes' rule

The model fits a Gaussian density to each class

New in version 017 QuadraticDiscriminantAnalysis

Read more in the User Guide

Parameters

1682 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

priors array optional shape nclasses Priors on classes

regparam float optional Regularizes the covariance estimate as

1regparam Sigma regparam npeyenfeatures

storecovariance boolean If True the covariance matrices are computed and stored in the selfcovariance attribute

New in version 017

tolfloat optional default 10e4 Threshold used for rank estimation

New in version 017

Attributes

covariance list of arraylike shape nfeatures nfeatures Covariance matrices of each class

means arraylike shape nclasses nfeatures Class means

priors arraylike shape nclasses Class priors sum to 1

rotations list of arrays For each class k an array of shape nfeatures nk with nk minnfeatures number of elements in class k It is the rotation of the Gaussian distribution ie its principal axis

scalings list of arrays For each class k an array of shape nk It contains the scaling of the Gaussian distributions along its principal axes ie the variance in the rotated coordinate system

See also

sklearndiscriminantanalysisLinearDiscriminantAnalysis Linear Discriminant Analysis

Examples

```
from sklearndiscriminantanalysis import QuadraticDiscriminantAnalysis
import numpy as np
X nparray1 1 2 1 3 2 1 1 2 1 3 2
y nparray1 1 1 2 2 2
clf QuadraticDiscriminantAnalysis
clffitX y
```

QuadraticDiscriminantAnalysispriorsNone regparam00

storecovarianceFalse tol00001

```
printclfpredict08 1
1
```

Methods

decisionfunction self X Apply decision function to an array of samples

fitself X y Fit the model according to the given training data and parameters

getparams self deep Get parameters for this estimator

Continued on next page

610sklearndiscriminantanalysis Discriminant Analysis 1683

`predict` self X Perform classification on an array of test vectors X

`predictlogproba` self X Return posterior probabilities of classification

`predictproba` self X Return posterior probabilities of classification

`score` self X y sampleweight Returns the mean accuracy on the given test data and labels

`setparams` self params Set the parameters of this estimator

`init` selfpriorsNone regparam00 storecovarianceFalse tol000001

`decisionfunction` selfX

Apply decision function to an array of samples

Parameters

Xarraylike shape nsamples nfeatures Array of samples test vectors

Returns

Carray shape nsamples nclasses or nsamples Decision function values related to each class per sample In the twoclass case the shape is nsamples giving the log likelihood ratio of the positive class

`fitselfXy`

Fit the model according to the given training data and parameters

Changed in version 019 storecovariances has been moved to main constructor as storecovariance

Changed in version 019 tol has been moved to main constructor

Parameters

Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

yarray shape nsamples Target values integers

`getparams` selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

`predictselfX`

Perform classification on an array of test vectors X

The predicted class C for each sample in X is returned

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carray shape nsamples

scikitlearn user guide Release 0213

predictlogproba selfX

Return posterior probabilities of classification

Parameters

Xarraylike shape nsamples nfeatures Array of samplestest vectors

Returns

Carray shape nsamples nclasses Posterior logprobabilities of classification per class

predictproba selfX

Return posterior probabilities of classification

Parameters

Xarraylike shape nsamples nfeatures Array of samplestest vectors

Returns

Carray shape nsamples nclasses Posterior probabilities of classification per class

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearndiscriminantanalysisQuadraticDiscriminantAnalysis

- Classifier comparison

- Linear and Quadratic Discriminant Analysis with covariance ellipsoid

611sklearndummy Dummy estimators

User guide See the Model evaluation quantifying the quality of predictions section for further details

611sklearndummy Dummy estimators 1685

scikitlearn user guide Release 0213

dummyDummyClassifier strategy DummyClassifier is a classifier that makes predictions using simple rules

dummyDummyRegressor strategy constant DummyRegressor is a regressor that makes predictions using simple rules

6111sklearndummy DummyClassifier

classsklearndummy DummyClassifier strategy'stratifed' randomstateNone constantNone DummyClassifier is a classifier that makes predictions using simple rules

This classifier is useful as a simple baseline to compare with other real classifiers Do not use it for real problems

Read more in the User Guide

Parameters

strategy str default"stratifed" Strategy to use to generate predictions

- "stratifed" generates predictions by respecting the training set's class distribution
 - "mostfrequent" always predicts the most frequent label in the training set
 - "prior" always predicts the class that maximizes the class prior like "mostfrequent"
- andpredictproba returns the class prior

- "uniform" generates predictions uniformly at random

- "constant" always predicts a constant label that is provided by the user This is useful for metrics that evaluate a nonmajority class

New in version 017 Dummy Classifier now supports prior fitting strategy using parameterprior

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

constant int or str or array of shape noutputs The explicit constant as predicted by the "constant" strategy This parameter is useful only for the "constant" strategy

Attributes

classes array or list of array of shape nclasses Class labels for each output

nclasses array or list of array of shape nclasses Number of label for each output

classprior array or list of array of shape nclasses Probability of each class for each output

noutputs int Number of outputs

sparseoutput bool True if the array returned from predict is to be in sparse CSC format Is automatically set to True if the input y is passed in sparse format

Methods

1686 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

fitself X y sampleweight Fit the random classifier

getparams self deep Get parameters for this estimator

predict self X Perform classification on test vectors X

predictlogproba self X Return log probability estimates for the test vectors X

predictproba self X Return probability estimates for the test vectors X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init selfstrategy' stratified' randomstateNone constantNone

fitselfXysampleweightNone

Fit the random classifier

Parameters

Xarraylike object with finite length or shape Training data requires length nsamples

yarraylike shape nsamples or nsamples noutputs Target values

sampleweight arraylike of shape nsamples optional Sample weights

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Perform classification on test vectors X

Parameters

Xarraylike object with finite length or shape Training data requires length nsamples

Returns

yarray shape nsamples or nsamples noutputs Predicted target values for X

predictlogproba selfX

Return log probability estimates for the test vectors X

Parameters

Xarraylike object with finite length or shape Training data requires length nsamples

Returns

Parraylike or list of arraylike of shape nsamples nclasses Returns the log probability of the sample for each class in the model where classes are ordered arithmetically for each output

611sklearndummy Dummy estimators 1687

scikitlearn user guide Release 0213

predictproba selfX

Return probability estimates for the test vectors X

Parameters

Xarraylike object with finite length or shape Training data requires length

nsamples

Returns

Parraylike or list of arraylike of shape nsamples nclasses Returns the probability of the sample for each class in the model where classes are ordered arithmetically for each output

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike None Test samples with shape nsamples nfeatures or None Passing

None as test samples gives the same result as passing real test samples since DummyClassifier operates independently of the sampled observations

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

6112sklearnndummy DummyRegressor

classsklearnndummy DummyRegressor strategy'mean' constantNone quantileNone

DummyRegressor is a regressor that makes predictions using simple rules

This regressor is useful as a simple baseline to compare with other real regressors Do not use it for real problems

Read more in the User Guide

Parameters

strategy str Strategy to use to generate predictions

- "mean" always predicts the mean of the training set
- "median" always predicts the median of the training set
- "quantile" always predicts a specified quantile of the training set provided with the quantile parameter

1688 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- “constant” always predicts a constant value that is provided by the user
- constant int or float or array of shape noutputs The explicit constant as predicted by the “constant” strategy This parameter is useful only for the “constant” strategy
- quantile float in 00 10 The quantile to predict using the “quantile” strategy A quantile of 05 corresponds to the median while 00 to the minimum and 10 to the maximum

Attributes

constant float or array of shape noutputs Mean or median or quantile of the training targets or constant value given by the user

noutputs int Number of outputs

Methods

fitself X y sampleweight Fit the random regressor

getparams self deep Get parameters for this estimator

predict self X returnstd Perform classification on test vectors X

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfstrategy‘mean’ constantNone quantileNone

fitselfXysampleweightNone

Fit the random regressor

Parameters

Xarraylike object with finite length or shape Training data requires length

nsamples

yarraylike shape nsamples or nsamples noutputs Target values

sampleweight arraylike of shape nsamples optional Sample weights

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXreturnstdFalse

Perform classification on test vectors X

Parameters

Xarraylike object with finite length or shape Training data requires length

nsamples

611sklearndummy Dummy estimators 1689

scikitlearn user guide Release 0213

returnstd boolean optional Whether to return the standard deviation of posterior prediction All zeros in this case

Returns

yarray shape nsamples or nsamples noutputs Predicted target values for X

ystd array shape nsamples or nsamples noutputs Standard deviation of predictive distribution of query points

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike None Test samples with shape nsamples nfeatures or None For

some estimators this may be a precomputed kernel matrix instead shape nsamples

nsamplesfitted where nsamplesfitted is the number of samples used in the fitting

for the estimator Passing None as test samples gives the same result as passing real test

samples since DummyRegressor operates independently of the sampled observations

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

612sklearnensemble Ensemble Methods

Thesklearnensemble module includes ensemblebased methods for classification regression and anomaly de

tection

User guide See the Ensemble methods section for further details

ensembleAdaBoostClassifier An AdaBoost classifier

ensembleAdaBoostRegressor baseestimator

An AdaBoost regressor

ensembleBaggingClassifier baseestimator

A Bagging classifier

Continued on next page

1690 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Table 671 - continued from previous page

ensembleBaggingRegressor baseestimator

A Bagging regressor

ensembleExtraTreesClassifier An extratrees classifier

ensembleExtraTreesRegressor nestimators

An extratrees regressor

ensembleGradientBoostingClassifier loss

Gradient Boosting for classification

ensembleGradientBoostingRegressor loss

Gradient Boosting for regression

ensembleIsolationForest nestimators Isolation Forest Algorithm

ensembleRandomForestClassifier A random forest classifier

ensembleRandomForestRegressor A random forest regressor

ensembleRandomTreesEmbedding An ensemble of totally random trees

ensembleVotingClassifier estimators Soft VotingMajority Rule classifier for unfitted estimators

ensembleVotingRegressor estimators Prediction voting regressor for unfitted estimators

ensembleHistGradientBoostingRegressor Histogrambased Gradient Boosting Regression Tree

ensembleHistGradientBoostingClassifier Histogrambased Gradient Boosting Classification Tree

6121sklearnensemble AdaBoostClassifier

classsklearnensemble AdaBoostClassifier baseestimatorNone nestimators50 learn

ingrate10 algorithm'SAMMER' ran

domstateNone

An AdaBoost classifier

An AdaBoost 1 classifier is a metaestimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases

This class implements the algorithm known as AdaBoostSAMME 2

Read more in the User Guide

Parameters

baseestimator object optional defaultNone The base estimator from which the boosted

ensemble is built Support for sample weighting is required as well as proper

classes andnclasses attributes If None then the base estimator is

DecisionTreeClassifiermaxdepth1

nestimators integer optional default50 The maximum number of estimators at which

boosting is terminated In case of perfect fit the learning procedure is stopped early

learningrate float optional default1 Learning rate shrinks the contribution of each

classifier by learningrate There is a tradeoff between learningrate and

nestimators

algorithm 'SAMME' 'SAMMER' optional default'SAMMER' If 'SAMMER' then

use the SAMMER real boosting algorithm baseestimator must support calculation

of class probabilities If 'SAMME' then use the SAMME discrete boosting algorithm The

SAMMER algorithm typically converges faster than SAMME achieving a lower test error

with fewer boosting iterations

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

612sklearnensemble Ensemble Methods 1691

scikitlearn user guide Release 0213

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Attributes

estimators list of classifiers The collection of fitted subestimators

classes array of shape nclasses The classes labels

nclasses int The number of classes

estimatorweights array of floats Weights for each estimator in the boosted ensemble

estimatorerrors array of floats Classification error for each estimator in the boosted ensemble

featureimportances array of shape nfeatures Return the feature importances

the higher the more important the feature

See also

AdaBoostRegressor GradientBoostingClassifier

sklearn.tree.DecisionTreeClassifier

References

R33e4ec8c4ad51 R33e4ec8c4ad52

Examples

```
from sklearn.ensemble import AdaBoostClassifier
```

```
from sklearn.datasets import make_classification
```

```
X, y = make_classification(n_samples=1000, n_features=4,
```

```
                           n_informative=2, n_redundant=0,
```

```
                           random_state=0, shuffle=False)
```

```
clf = AdaBoostClassifier(n_estimators=100, random_state=0)
```

```
clf.fit(X, y)
```

```
AdaBoostClassifier(algorithm='SAMME', base_estimator=None,
```

```
                    learning_rate=10, n_estimators=100, random_state=0)
```

```
clf.feature_importances_
```

```
array(0.28, 0.42, 0.14, 0.16)
```

```
clf.predict(0, 0, 0, 0)
```

```
array(1)
```

```
clf.score(X, y)
```

```
0.983
```

Methods

decisionfunction(self, X) Compute the decision function of X

fit(self, X, y, sample_weight) Build a boosted classifier from the training set X, y

get_params(self, deep=True) Get parameters for this estimator

predict(self, X) Predict classes for X

predict_log_proba(self, X) Predict class logprobabilities for X

Continued on next page

1692 Chapter 6 API (Application Programming Interface) Reference

`predictproba` self X Predict class probabilities for X

`score` self X y `sampleweight` Returns the mean accuracy on the given test data and labels

`setparams` self `params` Set the parameters of this estimator

`stageddecisionfunction` self X Compute decision function of Xfor each boosting iteration

`stagedpredict` self X Return staged predictions for X

`stagedpredictproba` self X Predict class probabilities for X

`stagedscore` self X y `sampleweight` Return staged scores for X y

`init` self`baseestimator``None` `nestimators``50` `learningrate``10` `algorithm`‘SAMMER’

`randomstate``None`

`decisionfunction` selfX

Compute the decision function of X

Parameters

Xarraylike sparse matrix of shape `nsamples nfeatures` The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

Returns

score array shape `nsamples k` The decision function of the input samples The order of outputs is the same of that of the classes attribute Binary classification is a special cases with`k = 1` otherwise `knclasses` For binary classification values closer to 1 or -1 mean more like the first or second class in `classes` respectively

`featureimportances`

Return the feature importances the higher the more important the feature

Returns

`featureimportances` array shape `nfeatures`

`fitselfXysampleweight``None`

Build a boosted classifier from the training set X y

Parameters

Xarraylike sparse matrix of shape `nsamples nfeatures` The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

yarraylike of shape `nsamples` The target values class labels

`sampleweight` arraylike of shape `nsamples` optional Sample weights If `None` the sample weights are initialized to 1 `nsamples`

Returns

self object

`getparams` self`deep``True`

Get parameters for this estimator

Parameters

612sklearnensemble Ensemble Methods 1693

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict classes for X

The predicted class of an input sample is computed as the weighted mean prediction of the classifiers in the ensemble

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

Returns

yarray of shape nsamples The predicted classes

predictlogproba selfX

Predict class logprobabilities for X

The predicted class logprobabilities of an input sample is computed as the weighted mean predicted class logprobabilities of the classifiers in the ensemble

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

Returns

parray of shape nsamples nclasses The class probabilities of the input samples The order of outputs is the same of that of the classes attribute

predictproba selfX

Predict class probabilities for X

The predicted class probabilities of an input sample is computed as the weighted mean predicted class probabilities of the classifiers in the ensemble

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

Returns

parray of shape nsamples nclasses The class probabilities of the input samples The order of outputs is the same of that of the classes attribute

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

1694 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

stageddecisionfunction selfX

Compute decision function of Xfor each boosting iteration

This method allows monitoring ie determine error on testing set after each boosting iteration

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input sam

ples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are

converted to CSR (Compressed Sparse Row)

Returns

score generator of array shape nsamples k The decision function of the input sam

ples The order of outputs is the same of that of the classes attribute Binary classification

is a special cases with k 1 otherwise knclasses For binary classification

values closer to 1 or 1 mean more like the first or second class in classes respectively

stagedpredict selfX

Return staged predictions for X

The predicted class of an input sample is computed as the weighted mean prediction of the classifiers in

the ensemble

This generator method yields the ensemble prediction after each iteration of boosting and therefore allows

monitoring such as to determine the prediction on a test set after each boost

Parameters

Xarraylike of shape nsamples nfeatures The input samples Sparse matrix can be

CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

Returns

ygenerator of array shape nsamples The predicted classes

stagedpredictproba selfX

Predict class probabilities for X

The predicted class probabilities of an input sample is computed as the weighted mean predicted class

probabilities of the classifiers in the ensemble

This generator method yields the ensemble predicted class probabilities after each iteration of boosting

and therefore allows monitoring such as to determine the predicted class probabilities on a test set after

each boost

Parameters

612sklearnensemble Ensemble Methods 1695

scikitlearn user guide Release 0213

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

Returns
pgenerator of array shape nsamples The class probabilities of the input samples The order of outputs is the same of that of the classes attribute
stagedscore selfXysampleweightNone

Return staged scores for X y
This generator method yields the ensemble score after each iteration of boosting and therefore allows monitoring such as to determine the score on a test set after each boost

Parameters
Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)
yarraylike shape nsamples Labels for X
sampleweight arraylike shape nsamples optional Sample weights

Returns
zfloat
Examples using sklearnensembleAdaBoostClassifier
•Classifier comparison
•Twoclass AdaBoost
•Multiclass AdaBoosted Decision Trees
•Discrete versus Real AdaBoost
•Plot the decision surfaces of ensembles of trees on the iris dataset

6122sklearnensemble AdaBoostRegressor
classssklearnensemble AdaBoostRegressor baseestimatorNone nestimators50 learn
ingrate10 loss'linear' randomstateNone
An AdaBoost regressor

An AdaBoost 1 regressor is a metaestimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction As such subsequent regressors focus more on difficult cases
This class implements the algorithm known as AdaBoostR2 2

Read more in the User Guide
Parameters
baseestimator object optional defaultNone The base estimator from which the boosted ensemble is built Support for sample weighting is required If None then the base estimator isDecisionTreeRegressormaxdepth3
nestimators integer optional default50 The maximum number of estimators at which boosting is terminated In case of perfect fit the learning procedure is stopped early
1696 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

learningrate float optional default1 Learning rate shrinks the contribution of each regressor by learningrate There is a tradeoff between learningrate and nestimators

loss 'linear' 'square' 'exponential' optional default'linear' The loss function to use when updating the weights after each boosting iteration

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Attributes

estimators list of classifiers The collection of fitted subestimators

estimatorweights array of floats Weights for each estimator in the boosted ensemble

estimatorerrors array of floats Regression error for each estimator in the boosted ensemble

featureimportances array of shape nfeatures Return the feature importances the higher the more important the feature

See also

AdaBoostClassifier GradientBoostingRegressor

sklearn.tree.DecisionTreeRegressor

References

R0c261b7dee9d1 R0c261b7dee9d2

Examples

```
from sklearn.ensemble import AdaBoostRegressor
```

```
from sklearn.datasets import make_regression
```

```
X, y = make_regression(n_features=4, n_informative=2,
```

```
random_state=0, shuffle=False)
```

```
regr = AdaBoostRegressor(random_state=0, n_estimators=100,
```

```
random_state=0)
```

```
regr.fit(X, y)
```

```
regr.feature_importances_
```

```
array([0.2788, 0.7109, 0.0065, 0.0036])
```

```
regr.predict([0, 0, 0, 0])
```

```
array([4.7972])
```

```
regr.score(X, y)
```

```
0.9771
```

Methods

fitself X y sampleweight Build a boosted regressor from the training set X y

Continued on next page

612sklearn.ensemble Ensemble Methods 1697

scikitlearn user guide Release 0213

Table 673 – continued from previous page

getparams self deep Get parameters for this estimator

predict self X Predict regression value for X

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

stagedpredict self X Return staged predictions for X

stagedscore self X y sampleweight Return staged scores for X y

init selfbaseestimatorNone nestimators50 learningrate10 loss'linear' randomstateNone

featureimportances

Return the feature importances the higher the more important the feature Returns

featureimportances array shape nfeatures

fitselfXysampleweightNone

Build a boosted regressor from the training set X y

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

yarraylike of shape nsamples The target values real numbers

sampleweight arraylike of shape nsamples optional Sample weights If None the sample weights are initialized to 1 nsamples

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict regression value for X

The predicted regression value of an input sample is computed as the weighted median prediction of the classifiers in the ensemble

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

Returns

1698 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

array of shape nsamples The predicted regression values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

stagedpredict selfX

Return staged predictions for X

The predicted regression value of an input sample is computed as the weighted median prediction of the

classifiers in the ensemble

This generator method yields the ensemble prediction after each iteration of boosting and therefore allows

monitoring such as to determine the prediction on a test set after each boost

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input sam

ples

Returns

ygenerator of array shape nsamples The predicted regression values

612sklearnensemble Ensemble Methods 1699

scikitlearn user guide Release 0213
stagedscore selfXysampleweightNone

Return staged scores for X y

This generator method yields the ensemble score after each iteration of boosting and therefore allows monitoring such as to determine the score on a test set after each boost

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrix can be CSC CSR (Compressed Sparse Row) COO DOK or LIL COO DOK and LIL are converted to CSR (Compressed Sparse Row)

yarraylike shape nsamples Labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

zfloat

Examples using sklearnensembleAdaBoostRegressor

•Decision Tree Regression with AdaBoost

6123sklearnensemble BaggingClassifier

classsklearnensemble BaggingClassifier baseestimatorNone nestimators10

maxsamples10 maxfeatures10 boot

strapTrue bootstrapfeaturesFalse

oobscoreFalse warmstartFalse njobsNone

randomstateNone verbose0

A Bagging classifier

A Bagging classifier is an ensemble metaestimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions either by voting or by averaging to form a final prediction Such a metaestimator can typically be used as a way to reduce the variance of a blackbox estimator eg a decision tree by introducing randomization into its construction procedure and then making an ensemble out of it

This algorithm encompasses several works from the literature When random subsets of the dataset are drawn as random subsets of the samples then this algorithm is known as Pasting Rb1846455d0e51 If samples are drawn with replacement then the method is known as Bagging Rb1846455d0e52 When random subsets of the dataset are drawn as random subsets of the features then the method is known as Random Subspaces Rb1846455d0e53 Finally when base estimators are built on subsets of both samples and features then the method is known as Random Patches Rb1846455d0e54

Read more in the User Guide

Parameters

baseestimator object or None optional defaultNone The base estimator to fit on random subsets of the dataset If None then the base estimator is a decision tree

nestimators int optional default10 The number of base estimators in the ensemble

maxsamples int or float optional default10 The number of samples to draw from X to train each base estimator

- If int then draw maxsamples samples
- If float then draw maxsamples Xshape0 samples

1700 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

maxfeatures int or float optional default10 The number of features to draw from X to train each base estimator

- If int then draw maxfeatures features
- If float then draw maxfeatures Xshape1 features

bootstrap boolean optional defaultTrue Whether samples are drawn with replacement If False sampling without replacement is performed

bootstrapfeatures boolean optional defaultFalse Whether features are drawn with replacement

oobscore bool optional defaultFalse Whether to use outofbag samples to estimate the generalization error

warmstart bool optional defaultFalse When set to True reuse the solution of the previous call to fit and add more estimators to the ensemble otherwise just fit a whole new ensemble See the Glossary

New in version 017 warmstart constructor parameter

njobs int or None optional defaultNone The number of jobs to run in parallel for both

fit and predict None means 1 unless in a joblibparallelbackend context

1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

verbose int optional default0 Controls the verbosity when fitting and predicting

Attributes

baseestimator estimator The base estimator from which the ensemble is grown

estimators list of estimators The collection of fitted base estimators

estimatorssamples list of arrays The subset of drawn samples for each base estimator

estimatorsfeatures list of arrays The subset of drawn features for each base estimator

classes array of shape nclasses The classes labels

nclasses int or list The number of classes

oobscore float Score of the training dataset obtained using an outofbag estimate

oobdecisionfunction array of shape nsamples nclasses Decision function computed

with outofbag estimate on the training set If nestimators is small it might

be possible that a data point was never left out during the bootstrap In this case

oobdecisionfunction might contain NaN

References

Rb1846455d0e51 Rb1846455d0e52 Rb1846455d0e53 Rb1846455d0e54

Methods

612sklearnensemble Ensemble Methods 1701

scikitlearn user guide Release 0213

decisionfunction self X Average of the decision functions of the base classifiers

fitself X y sampleweight Build a Bagging ensemble of estimators from the training set X y

getparams self deep Get parameters for this estimator

predict self X Predict class for X

predictlogproba self X Predict class logprobabilities for X

predictproba self X Predict class probabilities for X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init selfbaseestimatorNone nestimators10 maxsamples10 maxfeatures10 bootstrapTrue bootstrapfeaturesFalse oobscoreFalse warmstartFalse njobsNone

randomstateNone verbose0

decisionfunction selfX

Average of the decision functions of the base classifiers

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrices are accepted only if they are supported by the base estimator

Returns

score array shape nsamples k The decision function of the input samples The columns correspond to the classes in sorted order as they appear in the attribute classes Regression and binary classification are special cases with k 1 otherwiseknclasses

estimatorssamples

The subset of drawn samples for each base estimator

Returns a dynamically generated list of indices identifying the samples used for fitting each member of the ensemble ie the inbag samples

Note the list is recreated at each call to the property in order to reduce the object memory footprint by not storing the sampling data Thus fetching the property may be slower than expected

fitselfXysampleweightNone

Build a Bagging ensemble of estimators from the training set X y

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input samples Sparse matrices are accepted only if they are supported by the base estimator

yarraylike shape nsamples The target values class labels in classification real numbers in regression

sampleweight arraylike shape nsamples or None Sample weights If None then samples are equally weighted Note that this is supported only if the base estimator supports sample weighting

Returns

self object

1702 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class for X

The predicted class of an input sample is computed as the class with the highest mean predicted probability

If base estimators do not implement a predictproba method then it resorts to voting

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input sam

ples Sparse matrices are accepted only if they are supported by the base estimator

Returns

yarray of shape nsamples The predicted classes

predictlogproba selfX

Predict class logprobabilities for X

The predicted class logprobabilities of an input sample is computed as the log of the mean predicted class

probabilities of the base estimators in the ensemble

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input sam

ples Sparse matrices are accepted only if they are supported by the base estimator

Returns

parray of shape nsamples nclasses The class logprobabilities of the input samples

The order of the classes corresponds to that in the attribute classes

predictproba selfX

Predict class probabilities for X

The predicted class probabilities of an input sample is computed as the mean predicted class probabilities

of the base estimators in the ensemble If base estimators do not implement a predictproba method

then it resorts to voting and the predicted class probabilities of an input sample represents the proportion

of estimators predicting each class

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input sam

ples Sparse matrices are accepted only if they are supported by the base estimator

Returns

parray of shape nsamples nclasses The class probabilities of the input samples The

order of the classes corresponds to that in the attribute classes

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each

sample that each label set be correctly predicted

612sklearnensemble Ensemble Methods 1703

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

6124sklearnensemble BaggingRegressor

classsklearnensemble BaggingRegressor baseestimatorNone nestimators10

maxsamples10 maxfeatures10 boot

strapTrue bootstrapfeaturesFalse

oobscoreFalse warmstartFalse njobsNone

randomstateNone verbose0

A Bagging regressor

A Bagging regressor is an ensemble metaestimator that fits base regressors each on random subsets of the original dataset and then aggregate their individual predictions either by voting or by averaging to form a final prediction Such a metaestimator can typically be used as a way to reduce the variance of a blackbox estimator eg a decision tree by introducing randomization into its construction procedure and then making an ensemble out of it

This algorithm encompasses several works from the literature When random subsets of the dataset are drawn as random subsets of the samples then this algorithm is known as Pasting R4d113ba76fc01 If samples are drawn with replacement then the method is known as Bagging R4d113ba76fc02 When random subsets of the dataset are drawn as random subsets of the features then the method is known as Random Subspaces R4d113ba76fc03 Finally when base estimators are built on subsets of both samples and features then the method is known as Random Patches R4d113ba76fc04

Read more in the User Guide

Parameters

baseestimator object or None optional defaultNone The base estimator to fit on random subsets of the dataset If None then the base estimator is a decision tree

nestimators int optional default10 The number of base estimators in the ensemble

maxsamples int or float optional default10 The number of samples to draw from X to train each base estimator

- If int then draw maxsamples samples

- If float then draw maxsamples Xshape0 samples

maxfeatures int or float optional default10 The number of features to draw from X to train each base estimator

1704 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- If int then draw maxfeatures features
- If float then draw maxfeatures Xshape1 features

bootstrap boolean optional defaultTrue Whether samples are drawn with replacement If False sampling without replacement is performed

bootstrapfeatures boolean optional defaultFalse Whether features are drawn with replacement

oobscore bool Whether to use outofbag samples to estimate the generalization error

warmstart bool optional defaultFalse When set to True reuse the solution of the previous call to fit and add more estimators to the ensemble otherwise just fit a whole new ensemble See the Glossary

njobs int or None optional defaultNone The number of jobs to run in parallel for both fit and predict None means 1 unless in a joblibparallelbackend context

1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

verbose int optional default0 Controls the verbosity when fitting and predicting

Attributes

estimators list of estimators The collection of fitted subestimators

estimatorssamples list of arrays The subset of drawn samples for each base estimator

estimatorsfeatures list of arrays The subset of drawn features for each base estimator

oobscore float Score of the training dataset obtained using an outofbag estimate

oobprediction array of shape nsamples Prediction computed with outofbag estimate

on the training set If nestimators is small it might be possible that a data point was never left out during the bootstrap In this case oobprediction might contain NaN

References

R4d113ba76fc01 R4d113ba76fc02 R4d113ba76fc03 R4d113ba76fc04

Methods

fitself X y sampleweight Build a Bagging ensemble of estimators from the train

ing set X y

getparams self deep Get parameters for this estimator

predict self X Predict regression target for X

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

612sklearnensemble Ensemble Methods 1705

scikitlearn user guide Release 0213

init selfbaseestimatorNone nestimators10 maxsamples10 maxfeatures10 boot
strapTrue bootstrapfeaturesFalse oobscoreFalse warmstartFalse njobsNone
randomstateNone verbose0

estimatorssamples

The subset of drawn samples for each base estimator

Returns a dynamically generated list of indices identifying the samples used for fitting each member of the ensemble ie the inbag samples

Note the list is recreated at each call to the property in order to reduce the object memory footprint by not storing the sampling data Thus fetching the property may be slower than expected

fitselfXysampleweightNone

Build a Bagging ensemble of estimators from the training set X y

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input sam
ples Sparse matrices are accepted only if they are supported by the base estimator

yarraylike shape nsamples The target values class labels in classification real num
bers in regression

sampleweight arraylike shape nsamples or None Sample weights If None then
samples are equally weighted Note that this is supported only if the base estimator sup
ports sample weighting

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict regression target for X

The predicted regression target of an input sample is computed as the mean predicted regression targets of
the estimators in the ensemble

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The training input sam
ples Sparse matrices are accepted only if they are supported by the base estimator

Returns

yarray of shape nsamples The predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares ytrue ypred
2sum and v is the total sum of squares ytrue ytruemean 2sum The best possible score

1706 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may be a precomputed kernel matrix instead shape nsamples nsamplesfitted where nsamplesfitted is the number of samples used in the fitting for the estimator yarraylike shape nsamples or nsamples noutputs True values for X sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnensembleBaggingRegressor

•Single estimator versus bagging biasvariance decomposition

6125sklearnensemble IsolationForest
classsklearnensemble IsolationForest nestimators100 maxsamples'auto' contamination'legacy' maxfeatures10 bootstrapFalse
njobsNone behaviour'old' randomstateNone
verbose0 warmstartFalse
Isolation Forest Algorithm

Return the anomaly score of each sample using the IsolationForest algorithm

The IsolationForest 'isolates' observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature Since recursive partitioning can be represented by a tree structure the number of splittings required to isolate a sample is equivalent to the path length from the root node to the terminating node This path length averaged over a forest of such random trees is a measure of normality and our decision function

612sklearnensemble Ensemble Methods 1707

scikitlearn user guide Release 0213

Random partitioning produces noticeably shorter paths for anomalies Hence when a forest of random trees collectively produce shorter path lengths for particular samples they are highly likely to be anomalies

Read more in the User Guide

New in version 018

Parameters

nestimators int optional default100 The number of base estimators in the ensemble

maxsamples int or float optional default"auto"

The number of samples to draw from X to train each base estimator

- If int then draw maxsamples samples
- If float then draw maxsamples Xshape0 samples
- If "auto" then maxsamplesmin256 nsamples

If maxsamples is larger than the number of samples provided all samples will be used for all trees no sampling

contamination float in 0 05 optional default01 The amount of contamination of the data set ie the proportion of outliers in the data set Used when fitting to define the threshold on the decision function If 'auto' the decision function threshold is determined as in the original paper

Changed in version 020 The default value of contamination will change from 01 in 020 toauto in 022

maxfeatures int or float optional default10 The number of features to draw from X to train each base estimator

- If int then draw maxfeatures features
- If float then draw maxfeatures Xshape1 features

bootstrap boolean optional defaultFalse If True individual trees are fit on random subsets of the training data sampled with replacement If False sampling without replacement is performed

njobs int or None optional defaultNone The number of jobs to run in parallel for both fit andpredict None means 1 unless in a joblibparallelbackend context

1means using all processors See Glossary for more details

behaviour str default'old' Behaviour of the decisionfunction which can be ei ther 'old' or 'new' Passing behaviournew makes the decisionfunction

change to match other anomaly detection algorithm API (Application Programming Interface) which will be the default be haviour in the future As explained in details in the offset attribute documentation

thedecisionfunction becomes dependent on the contamination parameter in such a way that 0 becomes its natural threshold to detect outliers

New in version 020 behaviour is added in 020 for backcompatibility purpose

Deprecated since version 020 behaviourold is deprecated in 020 and will not be possible in 022

Deprecated since version 022 behaviour parameter will be deprecated in 022 and re moved in 024

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

1708 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom
verbose int optional default0 Controls the verbosity of the tree building process
warmstart bool optional defaultFalse When set to True reuse the solution of the previous call to fit and add more estimators to the ensemble otherwise just fit a whole new forest See the Glossary

New in version 021

Attributes

estimators list of DecisionTreeClassifier The collection of fitted subestimators
estimatorssamples list of arrays The subset of drawn samples for each base estimator
maxsamples integer The actual number of samples
offset float Offset used to define the decision function from the raw scores We have the relation
decisionfunction scoresamples offset Assuming behaviour 'new' offset is defined as follows When the contamination parameter is set to "auto" the offset is equal to 0.5 as the scores of inliers are close to 0 and the scores of outliers are close to 1 When a contamination parameter different than "auto" is provided the offset is defined in such a way we obtain the expected number of outliers samples with decision function 0 in training Assuming the behaviour parameter is set to 'old' we always have offset 0.5 making the decision function independent from the contamination parameter

Notes

The implementation is based on an ensemble of ExtraTreeRegressor The maximum depth of each tree is set to $\lceil \log_2 n \rceil$ where n is the number of samples used to build the tree see Liu et al 2008 for more details

References

Rd7ae0a2ae6881 Rd7ae0a2ae6882

Methods

decisionfunction self X Average anomaly score of X of the base classifiers
fitself X y sampleweight Fit estimator
fitpredict self X y Performs fit on X and returns labels for X
getparams self deep Get parameters for this estimator
predict self X Predict if a particular sample is an outlier or not
scoresamples self X Opposite of the anomaly score defined in the original paper
setparams self params Set the parameters of this estimator
init selfnestimators100 maxsamples'auto' contamination'legacy' maxfeatures10
bootstrapFalse njobsNone behaviour'old' randomstateNone verbose0
warmstartFalse
612sklearnensemble Ensemble Methods 1709

scikitlearn user guide Release 0213

decisionfunction selfX

Average anomaly score of X of the base classifiers

The anomaly score of an input sample is computed as the mean anomaly score of the trees in the forest

The measure of normality of an observation given a tree is the depth of the leaf containing this observation

which is equivalent to the number of splittings required to isolate this point In case of several observations

nleft in the leaf the average path length of a nleft samples isolation tree is added

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally

it will be converted to dtype=np.float32 and if a sparse matrix is provided to a sparse

csrmatrix

Returns

scores array shape nsamples The anomaly score of the input samples The lower the

more abnormal Negative scores represent outliers positive scores represent inliers

estimatorssamples

The subset of drawn samples for each base estimator

Returns a dynamically generated list of indices identifying the samples used for fitting each member of the

ensemble ie the inbag samples

Note the list is recreated at each call to the property in order to reduce the object memory footprint by

not storing the sampling data Thus fetching the property may be slower than expected

fitselfXyNone sampleweightNone

Fit estimator

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Use

dtype=np.float32 for maximum efficiency Sparse matrices are also supported use

sparse.csrmatrix for maximum efficiency

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted

ylgnored not used present for API (Application Programming Interface) consistency by convention

Returns

self object

fitpredict selfXyNone

Performs fit on X and returns labels for X

Returns 1 for outliers and 1 for inliers

Parameters

Xndarray shape nsamples nfeatures Input data

ylgnored not used present for API (Application Programming Interface) consistency by convention

Returns

yndarray shape nsamples 1 for inliers 1 for outliers

getparams selfdeepTrue

Get parameters for this estimator

Parameters

1710 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict if a particular sample is an outlier or not

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally

it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparse

csr matrix

Returns

isinlier array shape nsamples For each observation tells whether or not 1 or 1 it

should be considered as an inlier according to the fitted model

scoresamples selfX

Opposite of the anomaly score defined in the original paper

The anomaly score of an input sample is computed as the mean anomaly score of the trees in the forest

The measure of normality of an observation given a tree is the depth of the leaf containing this observation

which is equivalent to the number of splittings required to isolate this point In case of several observations

nleft in the leaf the average path length of a nleft samples isolation tree is added

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples

Returns

scores array shape nsamples The anomaly score of the input samples The lower the

more abnormal

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnensembleIsolationForest

•Comparing anomaly detection algorithms for outlier detection on toy datasets

•IsolationForest example

612sklearnensemble Ensemble Methods 1711

scikitlearn user guide Release 0213

6126sklearnensemble RandomTreesEmbedding

classsklearnensemble RandomTreesEmbedding nestimators'warn' maxdepth5

minsamplessplit2 minsamplesleaf1

minweightfractionleaf00

maxleafnodesNone

minimpuritydecrease00

minimpuritysplitNone sparseoutputTrue

njobsNone randomstateNone verbose0

warmstartFalse

An ensemble of totally random trees

An unsupervised transformation of a dataset to a highdimensional sparse representation A datapoint is coded

according to which leaf of each tree it is sorted into Using a onehot encoding of the leaves this leads to a

binary coding with as many ones as there are trees in the forest

The dimensionality of the resulting representation is nout nestimators maxleafnodes

Ifmaxleafnodes None the number of leaf nodes is at most nestimators 2

maxdepth

Read more in the User Guide

Parameters

nestimators integer optional default10 Number of trees in the forest

Changed in version 020 The default value of nestimators will change from 10 in

version 020 to 100 in version 022

maxdepth integer optional default5 The maximum depth of each tree If None

then nodes are expanded until all leaves are pure or until all leaves contain less than

minsamplessplit samples

minsamplessplit int float optional default2 The minimum number of samples required

to split an internal node

- If int then consider minsamplessplit as the minimum number

- If float then minsamplessplit is a fraction and ceilminsamplessplit

nsamples is the minimum number of samples for each split

Changed in version 018 Added float values for fractions

minsamplesleaf int float optional default1 The minimum number of samples required

to be at a leaf node A split point at any depth will only be considered if it leaves at least

minsamplesleaf training samples in each of the left and right branches This may

have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number

- If float then minsamplesleaf is a fraction and ceilminsamplesleaf

nsamples is the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the

sum total of weights of all the input samples required to be at a leaf node Samples have

equal weight when sampleweight is not provided

maxleafnodes int or None optional defaultNone Grow trees with maxleafnodes

in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then

unlimited number of leaf nodes

1712 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

N_t N impurity N_tR N_t rightimpurity

N_tL N_t leftimpurity

where N is the total number of samples N_t is the number of samples at the current node

N_tL is the number of samples in the left child and N_tR is the number of samples in

the right child

NN_tN_tR and NN_tL all refer to the weighted sum if sampleweight is passed

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node

will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of

minimpuritydecrease in 019 The default value of minimpuritysplit

will change from 1e7 to 0 in 023 and it will be removed in 025 Use

minimpuritydecrease instead

sparseoutput bool optional defaultTrue Whether or not to return a sparse CSR (Compressed Sparse Row) matrix

as default behavior or to return a dense array compatible with dense pipeline operators

njobs int or None optional defaultNone The number of jobs to run in parallel for both

fit and predict None means 1 unless in a joblibparallelbackend context

1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

verbose int optional default0 Controls the verbosity when fitting and predicting

warmstart bool optional defaultFalse When set to True reuse the solution of the pre

vious call to fit and add more estimators to the ensemble otherwise just fit a whole new

forest See the Glossary

Attributes

estimators list of DecisionTreeClassifier The collection of fitted subestimators

References

R6e47e53bacbd1 R6e47e53bacbd2

Methods

apply self X Apply trees in the forest to X return leaf indices

decisionpath self X Return the decision path in the forest

fitself X y sampleweight Fit estimator

fittransform self X y sampleweight Fit estimator and transform dataset

Continued on next page

612sklearnensemble Ensemble Methods 1713

Table 677 – continued from previous page

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X Transform dataset

init selfnestimators'warn' maxdepth5 minsamplesplit2 minsamplesleaf1

minweightfractionleaf00 maxleafnodesNone minimpuritydecrease00

minimpuritysplitNone sparseoutputTrue njobsNone randomstateNone

verbose0 warmstartFalse

applyselfX

Apply trees in the forest to X return leaf indices

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Inter

nally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided

it will be converted into a sparse csr matrix

Returns

Xleaves arraylike shape nsamples nestimators For each datapoint x in X and for

each tree in the forest return the index of the leaf x ends up in

decisionpath selfX

Return the decision path in the forest

New in version 018

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Inter

nally its dtype will be converted to dtype npfloat32 If a sparse matrix is provided

it will be converted into a sparse csr matrix

Returns

indicator sparse csr array shape nsamples nnodes Return a node indicator matrix

where non zero elements indicates that the samples goes through the nodes

nnodesptr array of size nestimators 1 The columns from indica

tor nnodesptr in nodesptr i1 gives the indicator value for the ith estimator

featureimportances

Return the feature importances the higher the more important the feature

Returns

featureimportances array shape nfeatures The values of this array sum to 1 unless

all trees are single node trees consisting of only the root node in which case it will be an

array of zeros

fitselfXyNone sampleweightNone

Fit estimator

Parameters

Xarraylike or sparse matrix shapensamples nfeatures The input samples Use

dtype npfloat32 for maximum efficiency Sparse matrices are also supported use

sparse csr matrix for maximum efficiency

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples or None Sample weights If None then samples are equally weighted Splits that would create child nodes with net zero or negative weight are ignored while searching for a split in each node In the case of classification splits are also ignored if they would result in any single class carrying a negative weight in either child node

Returns

self object

fittransform selfXyNone sampleweightNone

Fit estimator and transform dataset

Parameters

Xarraylike or sparse matrix shapensamples nfeatures Input data used to build

forests Use dtype=np.float32 for maximum efficiency

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Splits that would create child nodes with net zero or negative

weight are ignored while searching for a split in each node In the case of classification

splits are also ignored if they would result in any single class carrying a negative

weight in either child node

Returns

Xtransformed sparse matrix shapensamples nout Transformed dataset

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Transform dataset

Parameters

Xarraylike or sparse matrix shapensamples nfeatures Input data to be transformed

Usedtype=np.float32 for maximum efficiency Sparse matrices are also supported

use sparsecsrmatrix for maximum efficiency

Returns

Xtransformed sparse matrix shapensamples nout Transformed dataset

612sklearnensemble Ensemble Methods 1715

scikitlearn user guide Release 0213

Examples using sklearnensembleRandomTreesEmbedding

- Hashing feature transformation using Totally Random Trees

- Feature transformations with ensembles of trees

- Manifold learning on handwritten digits Locally Linear Embedding Isomap

6127sklearnensemble VotingClassifier

classsklearnensemble VotingClassifier estimators voting‘hard’ weightsNone

njobsNone flattentransformTrue

Soft VotingMajority Rule classifier for unfitted estimators

New in version 017

Read more in the User Guide

Parameters

estimators list of string estimator tuples Invoking the fit method on the

VotingClassifier will fit clones of those original estimators that will be stored

in the class attribute selfestimators An estimator can be set to None ordrop

usingsetparams

voting str ‘hard’ ‘soft’ default‘hard’ If ‘hard’ uses predicted class labels for majority

rule voting Else if ‘soft’ predicts the class label based on the argmax of the sums of the pre

dicted probabilities which is recommended for an ensemble of wellcalibrated classifiers

weights arraylike shape nclassifiers optional default‘None’ Sequence of weights

float orint to weight the occurrences of predicted class labels hard voting or class

probabilities before averaging soft voting Uses uniform weights if None

njobs int or None optional defaultNone The number of jobs to run in parallel for fit

None means 1 unless in a joblibparallelbackend context1means using all

processors See Glossary for more details

flattentransform bool optional defaultTrue Affects shape of transform output only when

voting‘soft’ If voting‘soft’ and flattentransformTrue transform method returns ma

trix with shape nsamples nclassifiers nclasses If flattentransformFalse it returns

nclassifiers nsamples nclasses

Attributes

estimators list of classifiers The collection of fitted subestimators as defined in

estimators that are not None

namedestimators Bunch object a dictionary with attribute access Attribute to access any

fitted subestimators by name

New in version 020

classes arraylike shape npredictions The classes labels

See also

VotingRegressor Prediction voting regressor

1716 Chapter 6 API (Application Programming Interface) Reference

Examples

```
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier, VotingClassifier
clf1 = LogisticRegression(solver='lbfgs', multi_class='multinomial',
                           random_state=1)
clf2 = RandomForestClassifier(n_estimators=50, random_state=1)
clf3 = GaussianNB()
X = np.array([1, 2, 1, 3, 2, 1, 1, 2, 1, 3, 2])
y = np.array([1, 1, 1, 2, 2, 2])
ecf1 = VotingClassifier(estimators=
    [ ('clf1', clf1), ('rf', clf2), ('gnb', clf3) ], voting='hard')
ecf1.fit(X, y)
print(ecf1.predict(X))
# 1 1 1 2 2 2
np.array_equal(ecf1.named_estimators_[0].predict(X),
    ecf1.named_estimators_[0].predict(X))
True
ecf2 = VotingClassifier(estimators=
    [ ('clf1', clf1), ('rf', clf2), ('gnb', clf3) ], voting='soft')
ecf2.fit(X, y)
print(ecf2.predict(X))
# 1 1 1 2 2 2
ecf3 = VotingClassifier(estimators=
    [ ('clf1', clf1), ('rf', clf2), ('gnb', clf3) ], voting='soft', weights=[2, 1, 1],
    flatten_transform=True)
ecf3.fit(X, y)
print(ecf3.predict(X))
# 1 1 1 2 2 2
print(ecf3.transform(X).shape)
# 6 6
```

Methods

- `fit(self, X, y, sample_weight=None)` Fit the estimators
- `fit_transform(self, X, y)` Fit to data then transform it
- `get_params(self, deep=True)` Get the parameters of the ensemble estimator
- `predict(self, X)` Predict class labels for X
- `score(self, X, y, sample_weight=None)` Returns the mean accuracy on the given test data and labels
- `set_params(self, **params)` Setting the parameters for the ensemble estimator
- `transform(self, X)` Return class labels or probabilities for X for each estimator

init(self, estimators, voting='hard', weights=None, n_jobs=None, flatten_transform=True)
fit(self, X, y, sample_weight=None)
Fit the estimators

Parameters

612sklearn.ensemble Ensemble Methods 1717

scikitlearn user guide Release 0213

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target values

sampleweight arraylike shape nsamples or None Sample weights If None then samples are equally weighted Note that this is supported only if all underlying estimators support sample weights

Returns

self object

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get the parameters of the ensemble estimator

Parameters

deep bool Setting it to True gets the various estimators and the parameters of the estimators as well

predictselfX

Predict class labels for X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples

Returns

maj arraylike shape nsamples Predicted class labels

predictproba

Compute probabilities of possible outcomes for samples in X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples

Returns

avg arraylike shape nsamples nclasses Weighted average probability for each class

per sample

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

1718 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Setting the parameters for the ensemble estimator

Valid parameter keys can be listed with getparams

Parameters

params keyword arguments Specific parameters using eg

setparamsparameternamenewvalue In addition to setting the parameters of

the ensemble estimator the individual estimators of the ensemble estimator can also be

set or replaced by setting them to None

Examples

In this example the RandomForestClassifier is removed clf1 LogisticRegression clf2 Random

ForestClassifier eclf VotingClassifierestimators'lr' clf1 'rf' clf2 eclfsetparamsrfNone

transform selfX

Return class labels or probabilities for X for each estimator

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where

nsamples is the number of samples and nfeatures is the number of features

Returns

probabilitiesorlabels

Ifvotingsoft andflattentransformTrue returns arraylike of shape

nclassifiers nsamples nclasses being class probabilities calculated by each clas

sifier

Ifvotingsoft and flattentransformFalse arraylike of shape

nclassifiers nsamples nclasses

Ifvotinghard arraylike of shape nsamples nclassifiers being class labels

predicted by each classifier

Examples using sklearnensembleVotingClassifier

•Plot the decision boundaries of a VotingClassifier

•Plot class probabilities calculated by the VotingClassifier

6128sklearnensemble VotingRegressor

classsklearnensemble VotingRegressor estimators weightsNone njobsNone

Prediction voting regressor for unfitted estimators

New in version 021

612sklearnensemble Ensemble Methods 1719

scikitlearn user guide Release 0213

A voting regressor is an ensemble metaestimator that fits base regressors each on the whole dataset It then averages the individual predictions to form a final prediction

Read more in the User Guide

Parameters

estimators list of string estimator tuples Invoking the fit method on the VotingRegressor will fit clones of those original estimators that will be stored in the class attribute selfestimators An estimator can be set to None ordrop usingsetparams

weights arraylike shape nregressors optional default'None' Sequence of weights float orint to weight the occurrences of predicted values before averaging Uses uniform weights if None

njobs int or None optional defaultNone The number of jobs to run in parallel in None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

Attributes

estimators list of regressors The collection of fitted subestimators as defined in estimators that are not None namedestimators Bunch object a dictionary with attribute access Attribute to access any fitted subestimators by name

See also

VotingClassifier Soft V otingMajority Rule classifier

Examples

```
import numpy as np
from sklearnlinearmodel import LinearRegression
from sklearnensemble import RandomForestRegressor
from sklearnensemble import VotingRegressor

r1 = LinearRegression
r2 = RandomForestRegressor(n_estimators=10, random_state=1)
X = np.array([1, 2, 4, 3, 9, 4, 16, 5, 25, 6, 36])
y = np.array([6, 12, 20, 30, 42])
er = VotingRegressor([r1, r2])
print(er.fit(X, y).predict(X))
[33 57 118 197 28 403]
```

Methods

fitself X y sampleweight Fit the estimators
fittransform self X y Fit to data then transform it
getparams self deep Get the parameters of the ensemble estimator
predict self X Predict regression target for X
score self X y sampleweight Returns the coefficient of determination R2 of the prediction
setparams self params Setting the parameters for the ensemble estimator

Continued on next page

1720 Chapter 6 API (Application Programming Interface) Reference

transform self X Return predictions for X for each estimator

init selfestimators weightsNone njobsNone

fitselfXysampleweightNone

Fit the estimators

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where

nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target values

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Note that this is supported only if all underlying estimators

support sample weights

Returns

self object

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get the parameters of the ensemble estimator

Parameters

deep bool Setting it to True gets the various estimators and the parameters of the estimators

as well

predictselfX

Predict regression target for X

The predicted regression target of an input sample is computed as the mean predicted regression targets of

the estimators in the ensemble

Parameters

Xarraylike sparse matrix of shape nsamples nfeatures The input samples

Returns

yarray of shape nsamples The predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

612sklearnensemble Ensemble Methods 1721

scikitlearn user guide Release 0213

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may be a precomputed kernel matrix instead shape nsamples nsamplesfitted where nsamplesfitted is the number of samples used in the fitting for the estimator yarraylike shape nsamples or nsamples noutputs True values for X sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage setparams selfparams

Setting the parameters for the ensemble estimator

Valid parameter keys can be listed with getparams

Parameters

params keyword arguments Specific parameters using eg setparamsparameternamenewvalue In addition to setting the parameters of the ensemble estimator the individual estimators of the ensemble estimator can also be set or replaced by setting them to None

Examples

In this example the RandomForestClassifier is removed clf1 LogisticRegression clf2 RandomForestClassifier eclf VotingClassifierestimators'lr' clf1 'rf' clf2 eclfsetparamsrfNone

transform selfX

Return predictions for X for each estimator

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input samples

Returns

predictions arraylike of shape nsamples nclassifiers being values predicted by each

regressor

Examples using sklearnensembleVotingRegressor

•Plot individual and voting regression predictions

1722 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

6129sklearnensemble HistGradientBoostingRegressor

classsklearnensemble HistGradientBoostingRegressor loss'leastsquares' learn

ingrate01 maxiter100

maxleafnodes31

maxdepthNone

minsamplesleaf20

l2regularization00

maxbins256 scoringNone

validationfraction01

niternochangeNone

tol1e07 verbose0 ran

domstateNone

Histogrambased Gradient Boosting Regression Tree

This estimator is much faster than GradientBoostingRegressor for big datasets nsamples 10 000

The input data Xis prebinned into integervalued bins which considerably reduces the number of splitting

points to consider and allows the algorithm to leverage integerbased data structures For small sample sizes

GradientBoostingRegressor might be preferred since binning may lead to split points that are too

approximate in this setting

This implementation is inspired by LightGBM

Note This estimator is still experimental for now the predictions and the API (Application Programming Interface) might change without any

deprecation cycle To use it you need to explicitly import enablehistgradientboosting

explicitly require this experimental feature

from sklearnexperimental import enablehistgradientboosting noqa

now you can import normally from ensemble

from sklearnensemble import HistGradientBoostingClassifier

Parameters

loss 'leastsquares' optional default'leastsquares' The loss function to use in the boost

ing process Note that the "least squares" loss actually implements an "half least squares

loss" to simplify the computation of the gradient

learningrate float optional default01 The learning rate also known as shrinkage This

is used as a multiplicative factor for the leaves values Use 1for no shrinkage

maxiter int optional default100 The maximum number of iterations of the boosting pro

cess ie the maximum number of trees

maxleafnodes int or None optional default31 The maximum number of leaves for each

tree Must be strictly greater than 1 If None there is no maximum limit

maxdepth int or None optional defaultNone The maximum depth of each tree The

depth of a tree is the number of nodes to go from the root to the deepest leaf Must be

strictly greater than 1 Depth isn't constrained by default

minsamplesleaf int optional default20 The minimum number of samples per leaf For

small datasets with less than a few hundred samples it is recommended to lower this value

since only very shallow trees would be built

l2regularization float optional default0 The L2 regularization parameter Use 0for no

regularization default

612sklearnensemble Ensemble Methods 1723

scikitlearn user guide Release 0213

maxbins int optional default256 The maximum number of bins to use Before training each feature of the input array Xis binned into at most maxbins bins which allows for a much faster training stage Features with a small number of unique values may use less than maxbins bins Must be no larger than 256

scoring str or callable or None optional defaultNone Scoring parameter to use for early stopping It can be a single string see The scoring parameter defining model evaluation rules or a callable see Defining your scoring strategy from metric functions If None the estimator’s default scorer is used If scoringloss early stopping is checked wrt the loss value Only used if niternochange is not None

validationfraction int or float or None optional default01 Proportion or absolute size of training data to set aside as validation data for early stopping If None early stopping is done on the training data Only used if niternochange is not None

niternochange int or None optional defaultNone Used to determine when to “early stop” The fitting process is stopped when none of the last niternochange scores are better than the “niternochange 1”thtolast one up to some tolerance If None or 0 no earlystopping is done

tolfloat or None optional default1e7 The absolute tolerance to use when comparing scores during early stopping The higher the tolerance the more likely we are to early stop higher tolerance means that it will be harder for subsequent iterations to be considered an improvement upon the reference score

verbose int optional default0 The verbosity level If not zero print some information about the fitting process

randomstate int nprandomRandomStateInstance or None optional defaultNone Pseudorandom number generator to control the subsampling in the binning process and the trainvalidation data split if early stopping is enabled See randomstate

Attributes

niter int The number of iterations as selected by early stopping if niternochange is not None Otherwise it corresponds to maxiter

ntreesperiteration int The number of tree that are built at each iteration For regressors this is always 1

trainscore ndarray shape maxiter 1 The scores at each iteration on the training data The first entry is the score of the ensemble before the first iteration Scores are computed according to the scoring parameter If scoring is not ‘loss’ scores are computed on a subset of at most 10 000 samples Empty if no early stopping

validationsscore ndarray shape maxiter 1 The scores at each iteration on the held out validation data The first entry is the score of the ensemble before the first iteration Scores are computed according to the scoring parameter Empty if no early stopping or ifvalidationfraction is None

Examples

To use this experimental feature we need to explicitly ask for it

```
from sklearnexperimental import enablehistgradientboosting noqa
from sklearnensemble import HistGradientBoostingRegressor
from sklearndatasets import loadboston
X y loadbostonreturnXy True
est HistGradientBoostingRegressorfitX y
```

1724 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

estscoreX y
098

Methods

fitself X y Fit the gradient boosting model

getparams self deep Get parameters for this estimator

predict self X Predict values for X

score self X y sampleweight Returns the coefficient of determination R2 of the pre
diction

setparams self params Set the parameters of this estimator

init selfloss'leastquares' learningrate01 maxiter100 maxleafnodes31
maxdepthNone minsamplesleaf20 l2regularization00 maxbins256 scor
ingNone validationfraction01 niternochangeNone tol1e07 verbose0
randomstateNone

fitselfXy

Fit the gradient boosting model

Parameters

Xarraylike shapensamples nfeatures The input samples

yarraylike shapensamples Target values

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict values for X

Parameters

Xarraylike shape nsamples nfeatures The input samples

Returns

yndarray shape nsamples The predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always
predicts the expected value of y disregarding the input features would get a R2 score of 00

612sklearnensemble Ensemble Methods 1725

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may be a precomputed kernel matrix instead shape nsamples nsamplesfitted where nsamplesfitted is the number of samples used in the fitting for the estimator yarraylike shape nsamples or nsamples noutputs True values for X sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

61210sklearnensemble HistGradientBoostingClassifier
classsklearnensemble HistGradientBoostingClassifier loss'auto' learn
ingrate01 maxiter100
maxleafnodes31
maxdepthNone
minsamplesleaf20
l2regularization00
maxbins256 scoringNone
validationfraction01
niterunchangeNone
tol1e07 verbose0 ran
domstateNone

Histogrambased Gradient Boosting Classification Tree

This estimator is much faster than GradientBoostingClassifier for big datasets nsamples 10000 The input data Xis prebinned into integervalued bins which considerably reduces the number of splitting points to consider and allows the algorithm to leverage integerbased data structures For small sample sizes GradientBoostingClassifier might be preferred since binning may lead to split points that are too approximate in this setting

This implementation is inspired by LightGBM

1726 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Note This estimator is still experimental for now the predictions and the API (Application Programming Interface) might change without any deprecation cycle To use it you need to explicitly import enablehistgradientboosting

explicitly require this experimental feature
from sklearnexperimental import enablehistgradientboosting noqa
now you can import normally from ensemble
from sklearnensemble import HistGradientBoostingClassifier

Parameters

loss ‘auto’ ‘binarycrossentropy’ ‘categoricalcrossentropy’ optional default‘auto’

The loss function to use in the boosting process ‘binarycrossentropy’ also known as logistic loss is used for binary classification and generalizes to ‘categoricalcrossentropy’ for multiclass classification ‘auto’ will automatically choose either loss depending on the nature of the problem

learningrate float optional default01 The learning rate also known as shrinkage This is used as a multiplicative factor for the leaves values Use 1for no shrinkage

maxiter int optional default100 The maximum number of iterations of the boosting process ie the maximum number of trees for binary classification For multiclass classificationnnclasses trees per iteration are built

maxleafnodes int or None optional default31 The maximum number of leaves for each tree Must be strictly greater than 1 If None there is no maximum limit

maxdepth int or None optional defaultNone The maximum depth of each tree The depth of a tree is the number of nodes to go from the root to the deepest leaf Must be strictly greater than 1 Depth isn’t constrained by default

minsamplesleaf int optional default20 The minimum number of samples per leaf For small datasets with less than a few hundred samples it is recommended to lower this value since only very shallow trees would be built

l2regularization float optional default0 The L2 regularization parameter Use 0 for no regularization

maxbins int optional default256 The maximum number of bins to use Before training each feature of the input array Xis binned into at most maxbins bins which allows for a much faster training stage Features with a small number of unique values may use less than maxbins bins Must be no larger than 256

scoring str or callable or None optional defaultNone Scoring parameter to use for early stopping It can be a single string see The scoring parameter defining model evaluation rules or a callable see Defining your scoring strategy from metric functions If None the estimator’s default scorer is used If scoringloss early stopping is checked wrt the loss value Only used if niternochange is not None

validationfraction int or float or None optional default01 Proportion or absolute size of training data to set aside as validation data for early stopping If None early stopping is done on the training data

niternochange int or None optional defaultNone Used to determine when to “early stop” The fitting process is stopped when none of the last niternochange scores are better than the “niternochange 1”thtolast one up to some tolerance If None or 0 no earlystopping is done

612sklearnensemble Ensemble Methods 1727

scikitlearn user guide Release 0213

tolfloat or None optional default1e7 The absolute tolerance to use when comparing scores The higher the tolerance the more likely we are to early stop higher tolerance means that it will be harder for subsequent iterations to be considered an improvement upon the reference score

verbose int optional default0 The verbosity level If not zero print some information about the fitting process

randomstate int nprandomRandomStateInstance or None optional defaultNone Pseudorandom number generator to control the subsampling in the binning process and the trainvalidation data split if early stopping is enabled See randomstate

Attributes

niter int The number of estimators as selected by early stopping if niternochange is not None Otherwise it corresponds to maxiter

ntreesperiteration int The number of tree that are built at each iteration This is equal to 1 for binary classification and to nclasses for multiclass classification

trainscore ndarray shape maxiter 1 The scores at each iteration on the training data The first entry is the score of the ensemble before the first iteration Scores are computed according to the scoring parameter If scoring is not 'loss' scores are computed on a subset of at most 10 000 samples Empty if no early stopping

validation_score ndarray shape maxiter 1 The scores at each iteration on the held out validation data The first entry is the score of the ensemble before the first iteration Scores are computed according to the scoring parameter Empty if no early stopping or ifvalidationfraction is None

Examples

To use this experimental feature we need to explicitly ask for it

```
from sklearn.experimental import enablehistgradientboosting
noqa
from sklearn.ensemble import HistGradientBoostingRegressor
from sklearn.datasets import loadiris
X y = loadiris().returnXy
True
clf = HistGradientBoostingClassifier()
clf.fit(X y)
clf.score(X y)
10
```

Methods

decisionfunction self X Compute the decision function of X

fitself X y Fit the gradient boosting model

getparams self deep Get parameters for this estimator

predict self X Predict classes for X

predictproba self X Predict class probabilities for X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

1728 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

init self loss'auto' learningrate01 maxiter100 maxleafnodes31
maxdepthNone minsamplesleaf20 l2regularization00 maxbins256 scor
ingNone validationfraction01 niternochangeNone tol1e07 verbose0
randomstateNone

decisionfunction selfX

Compute the decision function of X

Parameters

Xarraylike shape nsamples nfeatures The input samples

Returns

decision ndarray shape nsamples or nsamples ntreesperiteration The raw pre
dicted values ie the sum of the trees leaves for each sample ntreesperiteration is
equal to the number of classes in multiclass classification

fitselfXy

Fit the gradient boosting model

Parameters

Xarraylike shapensamples nfeatures The input samples
yarraylike shapensamples Target values

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict classes for X

Parameters

Xarraylike shape nsamples nfeatures The input samples

Returns

yndarray shape nsamples The predicted classes

predictproba selfX

Predict class probabilities for X

Parameters

Xarraylike shape nsamples nfeatures The input samples

Returns

pndarray shape nsamples nclasses The class probabilities of the input samples

612sklearnensemble Ensemble Methods 1729

scikitlearn user guide Release 0213

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

613sklearnexceptions Exceptions and warnings

The sklearnexceptions module includes all custom warnings and error classes used across scikitlearn

exceptionsChangedBehaviorWarning Warning class used to notify the user of any change in the behavior

exceptionsConvergenceWarning Custom warning to capture convergence problems

exceptionsDataConversionWarning Warning used to notify implicit data conversions happening in the code

exceptionsDataDimensionalityWarning Custom warning to notify potential issues with data dimensionality

exceptionsEfficiencyWarning Warning used to notify the user of inefficient computation

exceptionsFitFailedWarning Warning class used if there is an error while fitting the estimator

exceptionsNotFittedError Exception class to raise if estimator is used before fitting

exceptionsNonBLASDotWarning Warning used when the dot operation does not use BLAS

exceptionsUndefinedMetricWarning Warning used when the metric is invalid

6131sklearnexceptions ChangedBehaviorWarning

class sklearnexceptions ChangedBehaviorWarning

Warning class used to notify the user of any change in the behavior

Changed in version 018 Moved from sklearnbase

Attributes

1730 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

args

Methods

withtraceback Exceptionwithtracebacktb - set selftraceback
to tb and return self

withtraceback

Exceptionwithtracebacktb - set selftraceback to tb and return self

6132sklearnexceptions ConvergenceWarning

classsklearnexceptions ConvergenceWarning

Custom warning to capture convergence problems

Changed in version 018 Moved from sklearnutils

Attributes

args

Methods

withtraceback Exceptionwithtracebacktb - set selftraceback
to tb and return self

withtraceback

Exceptionwithtracebacktb - set selftraceback to tb and return self

Examples using sklearnexceptionsConvergenceWarning

- Multiclass sparse logisitic regression on newgroups20
- Early stopping of Stochastic Gradient Descent
- Compare Stochastic learning strategies for MLPClassifier
- Feature discretization

6133sklearnexceptions DataConversionWarning

classsklearnexceptions DataConversionWarning

Warning used to notify implicit data conversions happening in the code

This warning occurs when some input data needs to be converted or interpreted in a way that may not match the
user's expectations

For example this warning may occur when the user

- passes an integer array to a function which expects float input and will convert the input

613sklearnexceptions Exceptions and warnings 1731

scikitlearn user guide Release 0213

- requests a noncopying operation but a copy is required to meet the implementation's datatype expectations
- passes an input whose shape can be interpreted ambiguously

Changed in version 018 Moved from sklearnutilsvalidation

Attributes

args

Methods

withtraceback Exceptionwithtracebacktb - set selftraceback

to tb and return self

withtraceback

Exceptionwithtracebacktb - set selftraceback to tb and return self

6134sklearnexceptions DataDimensionalityWarning

classsklearnexceptions DataDimensionalityWarning

Custom warning to notify potential issues with data dimensionality

For example in random projection this warning is raised when the number of components which quantifies the dimensionality of the target projection space is higher than the number of features which quantifies the dimensionality of the original source space to imply that the dimensionality of the problem will not be reduced

Changed in version 018 Moved from sklearnutils

Attributes

args

Methods

withtraceback Exceptionwithtracebacktb - set selftraceback

to tb and return self

withtraceback

Exceptionwithtracebacktb - set selftraceback to tb and return self

6135sklearnexceptions EfficiencyWarning

classsklearnexceptions EfficiencyWarning

Warning used to notify the user of inefficient computation

This warning notifies the user that the efficiency may not be optimal due to some reason which may be included as a part of the warning message This may be subclassed into a more specific Warning class

New in version 018

Attributes

1732 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

args

Methods

withtraceback Exceptionwithtracebacktb - set selftraceback

to tb and return self

withtraceback

Exceptionwithtracebacktb - set selftraceback to tb and return self

6136sklearnexceptions FitFailedWarning

classssklearnexceptions FitFailedWarning

Warning class used if there is an error while fitting the estimator

This Warning is used in meta estimators GridSearchCV and RandomizedSearchCV and the crossvalidation

helper function crossvalscore to warn when there is an error while fitting the estimator

Attributes

args

Examples

from sklearnmodelselection import GridSearchCV

from sklearnsvm import LinearSVC

from sklearnexceptions import FitFailedWarning

import warnings

warningssimplefilteralways FitFailedWarning

gs GridSearchCVLinearSVC C 1 2 errorscore0 cv2

X y 1 2 3 4 5 6 7 8 0 0 1 1

with warningscatchwarningsrecord Trueasw

try

gsfitX y This will raise a ValueError since C is 0

except ValueError

pass

printreprw1message

FitFailedWarningEstimator fit failed The score on this traintest

partition for these parameters will be set to 0000000

Details nValueError Penalty term must be positive got C2n

Changed in version 018 Moved from sklearncrossvalidation

Methods

withtraceback Exceptionwithtracebacktb - set selftraceback

to tb and return self

613sklearnexceptions Exceptions and warnings 1733

scikitlearn user guide Release 0213

withtraceback

Exceptionwithtracebacktb - set selftraceback to tb and return self

6137sklearnexceptions NotFittedError

classsklearnexceptions NotFittedError

Exception class to raise if estimator is used before fitting

This class inherits from both ValueError and AttributeError to help with exception handling and backward compatibility

Attributes

args

Examples

```
from sklearnsvm import LinearSVC
```

```
from sklearnexceptions import NotFittedError
```

```
try
```

```
LinearSVCpredict1 2 2 3 3 4
```

```
except NotFittedError ase
```

```
printrepre
```

NotFittedErrorThis LinearSVC instance is not fitted yet

Changed in version 018 Moved from sklearnutilsvalidation

Methods

withtraceback Exceptionwithtracebacktb - set selftraceback to tb and return self

withtraceback

Exceptionwithtracebacktb - set selftraceback to tb and return self

6138sklearnexceptions NonBLASDotWarning

classsklearnexceptions NonBLASDotWarning

Warning used when the dot operation does not use BLAS

This warning is used to notify the user that BLAS was not used for dot operation and hence the efficiency may be affected

Changed in version 018 Moved from sklearnutilsvalidation extends EfficiencyWarning

Attributes

args

1734 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Methods

withtraceback Exceptionwithtracebacktb – set selftraceback to tb and return self

withtraceback

Exceptionwithtracebacktb – set selftraceback to tb and return self

6139sklearnexceptions UndefinedMetricWarning

classsklearnexceptions UndefinedMetricWarning

Warning used when the metric is invalid

Changed in version 018 Moved from sklearnbase

Attributes

args

Methods

withtraceback Exceptionwithtracebacktb – set selftraceback to tb and return self

withtraceback

Exceptionwithtracebacktb – set selftraceback to tb and return self

614sklearnexperimental Experimental

The sklearnexperimental module provides importable modules that enable the use of experimental features or estimators

The features and estimators that are experimental aren’t subject to deprecation cycles Use them at your own risks

experimentalenablehistgradientboosting Enables histogrambased gradient boosting estimators

experimentalenableiterativeimputer Enables IterativeImputer

6141 sklearnexperimentalenablehistgradientboosting

Enables histogrambased gradient boosting estimators

The API (Application Programming Interface) and results of these estimators might change without any deprecation cycle

Importing this file dynamically sets the sklearnensembleHistGradientBoostingClassifier and sklearnensembleHistGradientBoostingRegressor as attributes of the ensemble module

explicitly require this experimental feature

from sklearnexperimental import enablehistgradientboosting noqa

now you can import normally from ensemble

614sklearnexperimental Experimental 1735

scikitlearn user guide Release 0213

```
from sklearnensemble import HistGradientBoostingClassifier
from sklearnensemble import HistGradientBoostingRegressor
```

The noqa comment can be removed it just tells linters like flake8 to ignore the import which appears as unused

6142 sklearnexperimentalenableiterativeimputer

Enables IterativeImputer

The API (Application Programming Interface) and results of this estimator might change without any deprecation cycle

Importing this file dynamically sets sklearnimputeliterativeimputer as an attribute of the impute mod

ule

explicitly require this experimental feature

```
from sklearnexperimental import enableiterativeimputer noqa
```

now you can import normally from impute

```
from sklearnimpute import IterativeImputer
```

615sklearnfeatureextraction Feature Extraction

The sklearnfeatureextraction module deals with feature extraction from raw data It currently includes

methods to extract features from text and images

User guide See the Feature extraction section for further details

featureextractionDictVectorizer dtype

Transforms lists of featurevalue mappings to vectors

featureextractionFeatureHasher Implements feature hashing aka the hashing trick

6151sklearnfeatureextraction DictVectorizer

class sklearnfeatureextraction DictVectorizer dtypeclass 'numpyfloat64' separa

tor'' sparseTrue sortTrue

Transforms lists of featurevalue mappings to vectors

This transformer turns lists of mappings dictlike objects of feature names to feature values into Numpy arrays

or scipy sparse matrices for use with scikitlearn estimators

When feature values are strings this transformer will do a binary onehot aka oneofK coding one boolean

valued feature is constructed for each of the possible string values that the feature can take on For instance a

feature "f" that can take on the values "ham" and "spam" will become two features in the output one signifying

"fham" the other "fspam"

However note that this transformer will only do a binary onehot encoding when feature values are of type

string If categorical features are represented as numeric values such as int the DictVectorizer can be followed

by sklearnpreprocessingOneHotEncoder to complete binary onehot encoding

Features that do not occur in a sample mapping will have a zero value in the resulting arraymatrix

Read more in the User Guide

Parameters

1736 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

dtype callable optional The type of feature values Passed to Numpy arrays
matrix constructors as the dtype argument

separator string optional Separator string used when constructing new features for onehot coding

sparse boolean optional Whether transform should produce scipysparse matrices True by default

sort boolean optional Whether featurenames and vocabulary should be sorted when fitting True by default

Attributes

vocabulary dict A dictionary mapping feature names to feature indices

featurenames list A list of length nfeatures containing the feature names eg “fham” and “fspam”

See also

FeatureHasher performs vectorization using only a hash function

sklearnpreprocessingOrdinalEncoder handles nominalcategorical features encoded as columns of arbitrary data types

Examples

```
from sklearnfeatureextraction import DictVectorizer
v = DictVectorizer(sparse=False)
D = {'foo': 1, 'bar': 2, 'foo': 3, 'baz': 1}
X = v.fit_transform(D)
X
array([[2, 0, 1],
       [0, 1, 3]])
v.inverse_transform(X)
array([[20, 10, 10, 10, 10]])
v.transform({'foo': 4, 'unseen': 3})
array([[0, 0, 4]])
```

Methods

fit(self, X, y) Learn a list of feature name - indices mappings

fit_transform(self, X, y) Learn a list of feature name - indices mappings and transform X

get_feature_names(self) Returns a list of feature names ordered by their indices

get_params(self, deep=True) Get parameters for this estimator

inverse_transform(self, X) dict type Transform array or sparse matrix X back to feature mappings

restrict(self, support_indices) Restrict the features to those in support using feature selection

set_params(self, params) Set the parameters of this estimator

Continued on next page

615sklearnfeatureextraction Feature Extraction 1737

transform self X Transform featurevalue dicts to array or sparse matrix

init selfdtypeclass ‘numpyfloat64’ separator’’ sparseTrue sortTrue
fitselfXyNone

Learn a list of feature name indices mappings

Parameters

XMapping or iterable over Mappings Dicts or Mappings from feature names arbitrary

Python objects to feature values strings or convertible to dtype

yignored

Returns

self

fittransform selfXyNone

Learn a list of feature name indices mappings and transform X

Like fitX followed by transformX but does not require materializing X in memory

Parameters

XMapping or iterable over Mappings Dicts or Mappings from feature names arbitrary

Python objects to feature values strings or convertible to dtype

yignored

Returns

Xarray sparse matrix Feature vectors always 2d

getfeaturenames self

Returns a list of feature names ordered by their indices

If oneofK coding is applied to categorical features this will include the constructed feature names but not the original ones

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfXdictdtypeclass ‘dict’

Transform array or sparse matrix X back to feature mappings

X must have been produced by this DictVectorizer’s transform or fittransform method it may only have passed through transformers that preserve the number of features and their order

In the case of onehotoneofK coding the constructed feature names and values are returned rather than the original ones

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Sample matrix

scikitlearn user guide Release 0213

dicttype callable optional Constructor for feature mappings Must conform to the col
lectionsMapping API (Application Programming Interface)

Returns

Dlist of dicttype objects length nsamples Feature mappings for the samples in X

restrict selfsupport indicesFalse

Restrict the features to those in support using feature selection

This function modifies the estimator inplace

Parameters

support arraylike Boolean mask or list of indices as returned by the getsupport member
of feature selectors

indices boolean optional Whether support is a list of indices

Returns

self

Examples

from sklearnfeatureextraction import DictVectorizer

from sklearnfeatureselection import SelectKBest chi2

v DictVectorizer

D foo 1 bar 2 foo 3 baz 1

X vfittransformD

support SelectKBestchi2 k2fitX 0 1

vgetfeaturenames

bar baz foo

vrestrictsupportgetsupport

DictVectorizerdtype separator sortTrue

sparseTrue

vgetfeaturenames

bar foo

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it's possible to update each component
of a nested object

Returns

self

transform selfX

Transform featurevalue dicts to array or sparse matrix

Named features not encountered during fit or fittransform will be silently ignored

Parameters

XMapping or iterable over Mappings length nsamples Dicts or Mappings from

feature names arbitrary Python objects to feature values strings or convertible to dtype

Returns

615sklearnfeatureextraction Feature Extraction 1739

scikitlearn user guide Release 0213

Xaarray sparse matrix Feature vectors always 2d

Examples using sklearnfeatureextractionDictVectorizer

•Column Transformer with Heterogeneous Data Sources

•FeatureHasher and DictVectorizer Comparison

6152sklearnfeatureextraction FeatureHasher

classsklearnfeatureextraction FeatureHasher nfeatures1048576 inputtype'dict'

dtypeclass 'numpyfloat64' alter

natesignTrue

Implements feature hashing aka the hashing trick

This class turns sequences of symbolic feature names strings into scipysparse matrices using a hash function to compute the matrix column corresponding to a name The hash function employed is the signed 32bit version of Murmurhash3

Feature names of type byte string are used asis Unicode strings are converted to UTF8 first but no Unicode normalization is done Feature values must be finite numbers

This class is a lowmemory alternative to DictVectorizer and CountVectorizer intended for largescale online learning and situations where memory is tight eg when running prediction code on embedded devices

Read more in the User Guide

Parameters

nfeatures integer optional The number of features columns in the output matrices Small numbers of features are likely to cause hash collisions but large numbers will cause larger coefficient dimensions in linear learners

inputtype string optional default "dict" Either "dict" the default to accept dictionaries over featurename value "pair" to accept pairs of featurename value or "string" to accept single strings featurename should be a string while value should be a number In the case of "string" a value of 1 is implied The featurename is hashed to find the appropriate column for the feature The value's sign might be flipped in the output but see nonnegative below

dtype numpy type optional default npfloat64 The type of feature values Passed to scipysparse matrix constructors as the dtype argument Do not set this to bool npboolean or any unsigned integer type

alternatesign boolean optional default True When True an alternating sign is added to the features as to approximately conserve the inner product in the hashed space even for small nfeatures This approach is similar to sparse random projection

See also

DictVectorizer vectorizes stringvalued features using a hash table

sklearnpreprocessingOneHotEncoder handles nominalcategorical features

1740 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

from sklearnfeatureextraction import FeatureHasher

h = FeatureHasher(nfeatures=10

D = {'dog': 1, 'cat': 2, 'elephant': 4, 'dog': 2, 'run': 5

f = h.transform(D)

f.toarray()

array([[0, 0, 4, 1, 0, 0, 0, 0, 0, 2],

[0, 0, 0, 2, 5, 0, 0, 0, 0, 0]])

Methods

fit(self, X, y=None)

Fit to data then transform it

get_params(self, deep=True) Get parameters for this estimator

set_params(self, **kwargs) Set the parameters of this estimator

transform(self, X) Transform a sequence of instances to a scipy.sparse matrix

init(self, nfeatures=1048576, input_type='dict', dtypeclass='numpy.float64', alter

natesign=True)

fit(self, X=None, y=None)

Noop

This method doesn't do anything. It exists purely for compatibility with the scikitlearn transformer API (Application Programming Interface)

Parameters

Xarraylike

Returns

self: FeatureHasher

fit_transform(self, X=None, y=None) fit_params

Fit to data then transform it

Fits transformer to X and y with optional parameters fit_params and returns a transformed version of X

Parameters

X: numpy array of shape (n_samples, n_features) Training set

y: numpy array of shape (n_samples,) Target values

Returns

X_new: numpy array of shape (n_samples, n_features_new) Transformed array

get_params(self, deep=True)

Get parameters for this estimator

Parameters

deep: boolean optional. If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

615sklearnfeatureextraction Feature Extraction 1741

scikitlearn user guide Release 0213

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfrawX

Transform a sequence of instances to a scipysparse matrix

Parameters

rawX iterable over raw features length nsamples Samples Each sample

must be iterable an eg a list or tuple containing generating feature names and option

ally values see the inputtype constructor argument which will be hashed rawX need

not support the len function so it can be the result of a generator nsamples is determined

on the fly

Returns

Xscipysparse matrix shape nsamples selfnfeatures Feature matrix for use with

estimators or further transformers

Examples using sklearnfeatureextractionFeatureHasher

•FeatureHasher and DictVectorizer Comparison

6153 From images

The sklearnfeatureextractionimage submodule gathers utilities to extract features from images

featureextractionimage

extractpatches2d Reshape a 2D image into a collection of patches

featureextractionimage

gridtograph nx nyGraph of the pixeltopixel connections

featureextractionimage

imgtograph img Graph of the pixeltopixel gradient connections

featureextractionimage

reconstructfrompatches2d Reconstruct the image from all of its patches

featureextractionimage

PatchExtractor Extracts patches from a collection of images

sklearnfeatureextractionimage extractpatches2d

sklearnfeatureextractionimage extractpatches2d image patchsize

maxpatchesNone ran

domstateNone

Reshape a 2D image into a collection of patches

The resulting patches are allocated in a dedicated array

1742 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

image array shape imageheight imagewidth or imageheight imagewidth

nchannels The original image data For color images the last dimension specifies the channel a RGB image would have nchannels3

patchsize tuple of ints patchheight patchwidth the dimensions of one patch

maxpatches integer or float optional default is None The maximum number of patches to extract If maxpatches is a float between 0 and 1 it is taken to be a proportion of the total number of patches

randomstate int RandomState instance or None optional defaultNone Pseudo number

generator state used for random sampling to use if maxpatches is not None If int

randomstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the

RandomState instance used by nprandom

Returns

patches array shape npatches patchheight patchwidth or npatches patchheight

patchwidth nchannels The collection of patches extracted from the image where

npatches is eithermaxpatches or the total number of patches that can be extracted

Examples

from sklearndatasets import loadsampleimage

from sklearnfeatureextraction import image

Use the array data from the first image in this dataset

oneimage loadsampleimagechinajpg

printImage shape formatoneimageshape

Image shape 427 640 3

patches imageextractpatches2doneimage 2 2

printPatches shape formatpatchesshape

Patches shape 272214 2 2 3

Here are just two of these patches

printpatches1

174 201 231

174 201 231

173 200 230

173 200 230

printpatches800

187 214 243

188 215 244

187 214 243

188 215 244

Examples using sklearnfeatureextractionimageextractpatches2d

•Online learning of a dictionary of parts of faces

•Image denoising using dictionary learning

615sklearnfeatureextraction Feature Extraction 1743

scikitlearn user guide Release 0213
sklearnfeatureextractionimage gridtograph
sklearnfeatureextractionimage gridtograph nx ny nz1
maskNone returnasclass
'scipysparsecoocomatrix'
dtypeclass 'int'
Graph of the pixeltopixel connections
Edges exist if 2 voxels are connected
Parameters
nx int Dimension in x axis
ny int Dimension in y axis
nz int optional default 1 Dimension in z axis
mask ndarray of booleans optional An optional mask of the image to consider only part of
the pixels
returnas npndarray or a sparse matrix class optional The class to use to build the returned
adjacency matrix
dtype dtype optional default int The data of the returned sparse matrix By default it is int
Notes
For scikitlearn versions 0141 and prior returnasnpndarray was handled by returning a dense npmatrix
instance Going forward npndarray returns an npndarray as expected
For compatibility user code relying on this method should wrap its calls in npasarray to avoid type issues
sklearnfeatureextractionimage imgtograph
sklearnfeatureextractionimage imgtograph imgmaskNone returnasclass
'scipysparsecoocomatrix'
dtypeNone
Graph of the pixeltopixel gradient connections
Edges are weighted with the gradient values
Read more in the User Guide
Parameters
img ndarray 2D or 3D 2D or 3D image
mask ndarray of booleans optional An optional mask of the image to consider only part of
the pixels
returnas npndarray or a sparse matrix class optional The class to use to build the returned
adjacency matrix
dtype None or dtype optional The data of the returned sparse matrix By default it is the
dtype of img
1744 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Notes

For scikitlearn versions 0141 and prior returnasnpndarray was handled by returning a dense npmatrix instance Going forward npndarray returns an npndarray as expected
For compatibility user code relying on this method should wrap its calls in npasarray to avoid type issues
sklearnfeatureextractionimage reconstructfrompatches2d
sklearnfeatureextractionimage reconstructfrompatches2d patches im
agesize
Reconstruct the image from all of its patches
Patches are assumed to overlap and the image is constructed by filling in the patches from left to right top to bottom averaging the overlapping regions
Read more in the User Guide

Parameters
patches array shape npatches patchheight patchwidth or npatches patchheight
patchwidth nchannels The complete set of patches If the patches contain colour
information channels are indexed along the last dimension RGB patches would have
nchannels3
imagesize tuple of ints imageheight imagewidth or imageheight imagewidth
nchannels the size of the image that will be reconstructed

Returns
image array shape imagesize the reconstructed image
Examples using sklearnfeatureextractionimagereconstructfrompatches2d
•Image denoising using dictionary learning
sklearnfeatureextractionimage PatchExtractor
classsklearnfeatureextractionimage PatchExtractor patchsizeNone
maxpatchesNone ran
domstateNone

Extracts patches from a collection of images
Read more in the User Guide
Parameters
patchsize tuple of ints patchheight patchwidth the dimensions of one patch
maxpatches integer or float optional default is None The maximum number of patches per
image to extract If maxpatches is a float in 0 1 it is taken to mean a proportion of the
total number of patches
randomstate int RandomState instance or None optional defaultNone If int ran
domstate is the seed used by the random number generator If RandomState instance
randomstate is the random number generator If None the random number generator is
the RandomState instance used by nprandom
615sklearnfeatureextraction Feature Extraction 1745

scikitlearn user guide Release 0213

Examples

```
from sklearn.datasets import load_sample_images
from sklearn.feature_extraction import image
Use the array data from the second image in this dataset
X = load_sample_images(images1)
print Image.shape, format X.shape
Image shape 427 640 3
pe = image.PatchExtractor(patchsize=2, 2)
pe.fit X
petrans = petransform X
print Patches.shape, format petrans.shape
Patches shape 545706 2 2
```

Methods

```
fit self X y Do nothing and return the estimator unchanged
get_params self deep Get parameters for this estimator
set_params self params Set the parameters of this estimator
transform self X Transforms the image samples in X into a matrix of
patch data
init self patchsize=None maxpatches=None randomstate=None
fit self X y None
Do nothing and return the estimator unchanged
```

This method is just there to implement the usual API (Application Programming Interface) and hence work in pipelines

Parameters

```
X arraylike shape nsamples nfeatures Training data
get_params self deep True
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
```

Returns

```
params mapping of string to any Parameter names mapped to their values
set_params self params
Set the parameters of this estimator
The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form component__parameter so that it's possible to update each component
of a nested object
Returns
self
transform self X
```

Transforms the image samples in X into a matrix of patch data
1746 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xarray shape nsamples imageheight imagewidth or nsamples imageheight
imagewidth nchannels Array of images from which to extract patches For
color images the last dimension specifies the channel a RGB image would have
nchannels3

Returns

patches array shape npatches patchheight patchwidth or npatches
patchheight patchwidth nchannels The collection of patches extracted from the im
ages where npatches is eithernsamples maxpatches or the total number
of patches that can be extracted

6154 From text

Thesklearnfeatureextractiontext submodule gathers utilities to build feature vectors from text doc
uments

featureextractiontext

CountVectorizer Convert a collection of text documents to a matrix of token
counts

featureextractiontext

HashingVectorizer Convert a collection of text documents to a matrix of token
occurrences

featureextractiontext

TfidfTransformer Transform a count matrix to a normalized tf or tfidf repre
sentation

featureextractiontext

TfidfVectorizer Convert a collection of raw documents to a matrix of TF
IDF features

sklearnfeatureextractiontext CountVectorizer

classsklearnfeatureextractiontext CountVectorizer input'content' encoding'utf
8' decodeerror'strict'

stripaccentsNone low

ercaseTrue preproces

sorNone tokenizerNone

stopwordsNone to

kenpattern'ubwwb'

ngramrange1 1 ana

lyzer'word' maxdf10

mindf1 maxfeaturesNone

vocabularyNone binaryFalse

dtypeclass 'numpyint64'

Convert a collection of text documents to a matrix of token counts

This implementation produces a sparse representation of the counts using scipysparsecsrmatrix

If you do not provide an apriori dictionary and you do not use an analyzer that does some kind of feature
selection then the number of features will be equal to the vocabulary size found by analyzing the data

Read more in the User Guide

Parameters

input string 'filename' 'file' 'content' If 'filename' the sequence passed as an argument to
fit is expected to be a list of filenames that need reading to fetch the raw content to analyze

615sklearnfeatureextraction Feature Extraction 1747

scikitlearn user guide Release 0213

If 'file' the sequence items must have a 'read' method filelike object that is called to fetch the bytes in memory

Otherwise the input is expected to be the sequence strings or bytes items are expected to be analyzed directly

encoding string 'utf8' by default If bytes or files are given to analyze this encoding is used to decode

decodeerror 'strict' 'ignore' 'replace' Instruction on what to do if a byte sequence is given to analyze that contains characters not of the given encoding By default it is 'strict' meaning that a UnicodeDecodeError will be raised Other values are 'ignore' and 'replace'

stripaccents 'ascii' 'unicode' None Remove accents and perform other character normalization during the preprocessing step 'ascii' is a fast method that only works on characters that have an direct ASCII mapping 'unicode' is a slightly slower method that works on any characters None default does nothing

Both 'ascii' and 'unicode' use NFKD normalization from unicodedatanormalize

lowercase boolean True by default Convert all characters to lowercase before tokenizing preprocessor callable or None default Override the preprocessing string transformation stage while preserving the tokenizing and ngrams generation steps

tokenizer callable or None default Override the string tokenization step while preserving the preprocessing and ngrams generation steps Only applies if analyzer 'word'

stopwords string 'english' list or None default If 'english' a builtin stop word list for English is used There are several known issues with 'english' and you should consider an alternative see Using stop words

If a list that list is assumed to contain stop words all of which will be removed from the resulting tokens Only applies if analyzer 'word'

If None no stop words will be used maxdf can be set to a value in the range 07 10

to automatically detect and filter stop words based on intra corpus document frequency of terms

tokenpattern string Regular expression denoting what constitutes a "token" only used if analyzer 'word' The default regexp select tokens of 2 or more alphanumeric

characters punctuation is completely ignored and always treated as a token separator

ngramrange tuple minn maxn The lower and upper boundary of the range of nvalues for different ngrams to be extracted All values of n such that minn ≤ n ≤ maxn will

be used

analyzer string 'word' 'char' 'charwb' or callable Whether the feature should be made of word or character ngrams Option 'charwb' creates character ngrams only from text inside word boundaries ngrams at the edges of words are padded with space

If a callable is passed it is used to extract the sequence of features out of the raw unprocessed input

Changed in version 021

Since v021 if input isfilename orfile the data is first read from the file and then passed to the given callable analyzer

maxdf float in range 00 10 or int default10 When building the vocabulary ignore terms that have a document frequency strictly higher than the given threshold corpusspecific stop

1748 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

words If float the parameter represents a proportion of documents integer absolute counts

This parameter is ignored if vocabulary is not None

mindf float in range 00 10 or int default1 When building the vocabulary ignore terms

that have a document frequency strictly lower than the given threshold This value is also

called cutoff in the literature If float the parameter represents a proportion of documents

integer absolute counts This parameter is ignored if vocabulary is not None

maxfeatures int or None defaultNone If not None build a vocabulary that only consider

the top maxfeatures ordered by term frequency across the corpus

This parameter is ignored if vocabulary is not None

vocabulary Mapping or iterable optional Either a Mapping eg a dict where keys are terms

and values are indices in the feature matrix or an iterable over terms If not given a vocabu

lary is determined from the input documents Indices in the mapping should not be repeated

and should not have any gap between 0 and the largest index

binary boolean defaultFalse If True all non zero counts are set to 1 This is useful for

discrete probabilistic models that model binary events rather than integer counts

dtype type optional Type of the matrix returned by fittransform or transform

Attributes

vocabulary dict A mapping of terms to feature indices

stopwords set Terms that were ignored because they either

- occurred in too many documents maxdf

- occurred in too few documents mindf

- were cut off by feature selection maxfeatures

This is only available if no vocabulary was given

See also

HashingVectorizer TfidfVectorizer

Notes

Thestopwords attribute can get large and increase the model size when pickling This attribute is provided

only for introspection and can be safely removed using delattr or set to None before pickling

Examples

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
corpus
```

```
This is the first document
```

```
This document is the second document
```

```
And this is the third one
```

```
Is this the first document
```

```
vectorizer = CountVectorizer
```

```
X = vectorizer.fit_transform(corpus)
```

```
print(vectorizer.get_feature_names())
```

```
and document first is one second the third this
```

615sklearn.feature_extraction Feature Extraction 1749

scikitlearn user guide Release 0213

printXtoarray

0 1 1 1 0 0 1 0 1

0 2 0 1 0 1 1 0 1

1 0 0 1 1 0 1 1 1

0 1 1 1 0 0 1 0 1

Methods

buildanalyzer self Return a callable that handles preprocessing and tokenization

buildpreprocessor self Return a function to preprocess the text before tokenization

buildtokenizer self Return a function that splits a string into a sequence of tokens

decode self doc Decode the input into a string of unicode symbols

fitself rawdocuments y Learn a vocabulary dictionary of all tokens in the raw documents

fittransform self rawdocuments y Learn the vocabulary dictionary and return term document matrix

getfeaturenames self Array mapping from feature integer indices to feature name

getparams self deep Get parameters for this estimator

getstopwords self Build or fetch the effective stop words list

inversetransform self X Return terms per document with nonzero entries in X

setparams self params Set the parameters of this estimator

transform self rawdocuments Transform documents to documentterm matrix

init selfinput'content' encoding'utf8' decodeerror'strict' stripaccentsNone

lowercaseTrue preprocessorNone tokenizerNone stopwordsNone to

kenpattern'ubwwb' ngramrange1 1analyzer'word' maxdf10

mindf1 maxfeaturesNone vocabularyNone binaryFalse dtypeclass

'numpyint64'

buildanalyzer self

Return a callable that handles preprocessing and tokenization

buildpreprocessor self

Return a function to preprocess the text before tokenization

buildtokenizer self

Return a function that splits a string into a sequence of tokens

decodeselfdoc

Decode the input into a string of unicode symbols

The decoding strategy depends on the vectorizer parameters

Parameters

doc string The string to decode

fitselfrawdocuments yNone

Learn a vocabulary dictionary of all tokens in the raw documents

Parameters

1750 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

rawdocuments iterable An iterable which yields either str unicode or file objects

Returns

self

fittransform selfrawdocuments yNone

Learn the vocabulary dictionary and return termdocument matrix

This is equivalent to fit followed by transform but more efficiently implemented

Parameters

rawdocuments iterable An iterable which yields either str unicode or file objects

Returns

Xarray nsamples nfeatures Documentterm matrix

getfeaturenames self

Array mapping from feature integer indices to feature name

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getstopwords self

Build or fetch the effective stop words list

inversetransform selfX

Return terms per document with nonzero entries in X

Parameters

Xarray sparse matrix shape nsamples nfeatures

Returns

Xinv list of arrays len nsamples List of arrays of terms

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfrawdocuments

Transform documents to documentterm matrix

Extract token counts out of raw text documents using the vocabulary fitted with fit or the one provided to

the constructor

Parameters

rawdocuments iterable An iterable which yields either str unicode or file objects

615sklearnfeatureextraction Feature Extraction 1751

scikitlearn user guide Release 0213

Returns

Xsparse matrix nsamples nfeatures Documentterm matrix

Examples using sklearnfeatureextractiontextCountVectorizer

- Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation
- Sample pipeline for text feature extraction and evaluation

sklearnfeatureextractiontext HashingVectorizer

classsklearnfeatureextractiontext HashingVectorizer input'content'

encoding'utf8' de

codeerror'strict'

stripaccentsNone low

ercaseTrue preproces

sorNone tokenizerNone

stopwordsNone to

kenpattern'ubwwb'

ngramrange1

1 analyzer'word'

nfeatures1048576 bi

naryFalse norm'l2'

alternatesignTrue

dtypeclass

'numpyfloat64'

Convert a collection of text documents to a matrix of token occurrences

It turns a collection of text documents into a scipysparse matrix holding token occurrence counts or binary occurrence information possibly normalized as token frequencies if norm'l1' or projected on the euclidean unit sphere if norm'l2'

This text vectorizer implementation uses the hashing trick to find the token string name to feature integer index mapping

This strategy has several advantages

- it is very low memory scalable to large datasets as there is no need to store a vocabulary dictionary in memory
- it is fast to pickle and unpickle as it holds no state besides the constructor parameters
- it can be used in a streaming partial fit or parallel pipeline as there is no state computed during fit

There are also a couple of cons vs using a CountVectorizer with an inmemory vocabulary

- there is no way to compute the inverse transform from feature indices to string feature names which can be a problem when trying to introspect which features are most important to a model
- there can be collisions distinct tokens can be mapped to the same feature index However in practice this is rarely an issue if nfeatures is large enough eg 2^{18} for text classification problems
- no IDF weighting as this would render the transformer stateful

The hash function employed is the signed 32bit version of Murmurhash3

Read more in the User Guide

Parameters

1752 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

input string 'filename' 'file' 'content' If 'filename' the sequence passed as an argument to fit is expected to be a list of filenames that need reading to fetch the raw content to analyze If 'file' the sequence items must have a 'read' method filelike object that is called to fetch the bytes in memory Otherwise the input is expected to be the sequence strings or bytes items are expected to be analyzed directly encoding string default 'utf8' If bytes or files are given to analyze this encoding is used to decode decodeerror 'strict' 'ignore' 'replace' Instruction on what to do if a byte sequence is given to analyze that contains characters not of the given encoding By default it is 'strict' meaning that a UnicodeDecodeError will be raised Other values are 'ignore' and 'replace'

stripaccents 'ascii' 'unicode' None Remove accents and perform other character normalization during the preprocessing step 'ascii' is a fast method that only works on characters that have an direct ASCII mapping 'unicode' is a slightly slower method that works on any characters None default does nothing

Both 'ascii' and 'unicode' use NFKD normalization from unicodedatanormalize lowercase boolean default True Convert all characters to lowercase before tokenizing preprocessor callable or None default Override the preprocessing string transformation stage while preserving the tokenizing and ngrams generation steps tokenizer callable or None default Override the string tokenization step while preserving the preprocessing and ngrams generation steps Only applies if analyzer 'word' stopwords string 'english' list or None default If 'english' a builtin stop word list for English is used There are several known issues with 'english' and you should consider an alternative see Using stop words

If a list that list is assumed to contain stop words all of which will be removed from the resulting tokens Only applies if analyzer 'word'

tokenpattern string Regular expression denoting what constitutes a "token" only used if analyzer 'word' The default regexp selects tokens of 2 or more alphanumeric characters punctuation is completely ignored and always treated as a token separator ngramrange tuple minn maxx default 1 1 The lower and upper boundary of the range of nvalues for different ngrams to be extracted All values of n such that minn n maxx will be used

analyzer string 'word' 'char' 'charwb' or callable Whether the feature should be made of word or character ngrams Option 'charwb' creates character ngrams only from text inside word boundaries ngrams at the edges of words are padded with space If a callable is passed it is used to extract the sequence of features out of the raw unprocessed input

Changed in version 021

Since v021 if input is filename or file the data is first read from the file and then passed to the given callable analyzer

nfeatures integer default 20 The number of features columns in the output matrices Small numbers of features are likely to cause hash collisions but large numbers will cause larger coefficient dimensions in linear learners 615sklearnfeatureextraction Feature Extraction 1753

scikitlearn user guide Release 0.21.3

binary boolean default False If True all non zero counts are set to 1 This is useful for discrete probabilistic models that model binary events rather than integer counts

norm 'l1' 'l2' or None optional Norm used to normalize term vectors None for no normalization

alternatesign boolean optional default True When True an alternating sign is added to the features as to approximately conserve the inner product in the hashed space even for small nfeatures This approach is similar to sparse random projection

New in version 0.19

dtype type optional Type of the matrix returned by fit_transform or transform

See also

CountVectorizer TfidfVectorizer

Examples

```
from sklearn.feature_extraction.text import HashingVectorizer
```

corpus

```
This is the first document
This document is the second document
And this is the third one
Is this the first document
```

vectorizer = HashingVectorizer(n_features=2 ** 4)

```
X = vectorizer.fit_transform(corpus)
print X.shape
4 16
```

Methods

build_analyzer self Return a callable that handles preprocessing and tokenization

build_preprocessor self Return a function to preprocess the text before tokenization

build_tokenizer self Return a function that splits a string into a sequence of tokens

decode self doc Decode the input into a string of unicode symbols

fit self X y Does nothing this transformer is stateless

fit_transform self X y Transform a sequence of documents to a document-term matrix

get_params self deep Get parameters for this estimator

get_stop_words self Build or fetch the effective stop words list

partial_fit self X y Does nothing this transformer is stateless

set_params self params Set the parameters of this estimator

transform self X Transform a sequence of documents to a document-term matrix

scikitlearn user guide Release 0213

init selfinput'content' encoding'utf8' decodeerror'strict' stripaccentsNone
lowercaseTrue preprocessorNone tokenizerNone stopwordsNone to
kenpattern'ubwwb' ngramrange1 1analyzer'word' nfeatures1048576
binaryFalse norm'l2' alternatesignTrue dtypeclass 'numpyfloat64'
buildanalyzer self
Return a callable that handles preprocessing and tokenization
buildpreprocessor self
Return a function to preprocess the text before tokenization
buildtokenizer self
Return a function that splits a string into a sequence of tokens
decodeselfdoc
Decode the input into a string of unicode symbols
The decoding strategy depends on the vectorizer parameters
Parameters
doc string The string to decode
fitselfXyNone
Does nothing this transformer is stateless
Parameters
Xarraylike shape nsamples nfeatures Training data
fittransform selfXyNone
Transform a sequence of documents to a documentterm matrix
Parameters
Xiterable over raw text documents length nsamples Samples Each sample must be a
text document either bytes or unicode strings file name or file object depending on the
constructor argument which will be tokenized and hashed
yany Ignored This parameter exists only for compatibility with sklearnpipelinePipeline
Returns
Xscipysparse matrix shape nsamples selfnfeatures Documentterm matrix
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values
getstopwords self
Build or fetch the effective stop words list
partialfit selfXyNone
Does nothing this transformer is stateless
This method is just there to mark the fact that this transformer can work in a streaming setup
Parameters
615sklearnfeatureextraction Feature Extraction 1755

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures Training data

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Transform a sequence of documents to a documentterm matrix

Parameters

Xiterable over raw text documents length nsamples Samples Each sample must be a text document either bytes or unicode strings file name or file object depending on the constructor argument which will be tokenized and hashed

Returns

Xscipysparse matrix shape nsamples selfnfeatures Documentterm matrix

Examples using sklearnfeatureextractiontextHashingVectorizer

- Outofcore classification of text documents
- Clustering text documents using kmeans
- Classification of text documents using sparse features

sklearnfeatureextractiontext TfidfTransformer

classsklearnfeatureextractiontext TfidfTransformer norm'l2' useidfTrue

smoothidfTrue sublin

eartfFalse

Transform a count matrix to a normalized tf or tfidf representation

Tf means termfrequency while tfidf means termfrequency times inverse documentfrequency This is a common term weighting scheme in information retrieval that has also found good use in document classification The goal of using tfidf instead of the raw frequencies of occurrence of a token in a given document is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus

The formula that is used to compute the tfidf for a term t of a document d in a document set is $tfidf(d, t) = tf(d, t) \cdot idf(t)$ and the idf is computed as $idf(t) = \log \frac{n}{dft + 1}$ if smoothidfFalse where n is the total number of documents in the document set and dft is the document frequency of t the document frequency is the number of documents in the document set that contain the term t The effect of adding "1" to the idf in the equation above is that terms with zero idf ie terms that occur in all documents in a training set will not be entirely ignored Note that the idf formula above differs from the standard textbook notation that defines the idf as $idf(t) = \log \frac{n}{dft}$

IfsmoothidfTrue the default the constant "1" is added to the numerator and denominator of the idf as if an extra document was seen containing every term in the collection exactly once which prevents zero divisions $idf(d, t) = \log \frac{1 + n}{1 + dft}$

1756 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Furthermore the formulas used to compute tf and idf depend on parameter settings that correspond to the SMART notation used in IR as follows

Tf is “n” natural by default “l” logarithmic when sublineartfTrue Idf is “t” when useidf is given “n” none otherwise Normalization is “c” cosine when norml2 “n” none when normNone

Read more in the User Guide

Parameters

norm ‘l1’ ‘l2’ or None optional default ‘l2’ Each output row will have unit norm either ‘l2’ Sum of squares of vector elements is 1 The cosine similarity between two vectors is their dot product when l2 norm has been applied ‘l1’ Sum of absolute values of vector elements is 1 See preprocessingnormalize

useidf boolean defaultTrue Enable inversedocumentfrequency reweighting

smoothidf boolean defaultTrue Smooth idf weights by adding one to document frequencies as if an extra document was seen containing every term in the collection exactly once Prevents zero divisions

sublineartf boolean defaultFalse Apply sublinear tf scaling ie replace tf with 1 logtf

Attributes

idf array shape nfeatures The inverse document frequency IDF vector only defined if useidf is True

References

R1b90ac3ca370Yates2011 R1b90ac3ca370MRS2008

Methods

fitself X y Learn the idf vector global term weights

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X copy Transform a count matrix to a tf or tfidf representation

init selfnorm ‘l2’ useidfTrue smoothidfTrue sublineartfFalse

fitselfXyNone

Learn the idf vector global term weights

Parameters

Xsparse matrix nsamples nfeatures a matrix of termtoken counts

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

615sklearnfeatureextraction Feature Extraction 1757

scikitlearn user guide Release 0213

numpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfXcopyTrue

Transform a count matrix to a tf or tfidf representation

Parameters

Xsparse matrix nsamples nfeatures a matrix of termtoken counts

copy boolean default True Whether to copy X and operate on the copy or perform inplace operations

Returns

vectors sparse matrix nsamples nfeatures

Examples using sklearnfeatureextractiontextTfidfTransformer

- Sample pipeline for text feature extraction and evaluation
- Clustering text documents using kmeans

1758 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearnfeatureextractiontext TfidfVectorizer

classsklearnfeatureextractiontext TfidfVectorizer input'content' encoding'utf8' decodeerror'strict'

stripaccentsNone low

ercaseTrue preproces

sorNone tokenizerNone ana

lyzer'word' stopwordsNone

tokenpattern'ubwwb'

ngramrange1 1

maxdf10 mindf1

maxfeaturesNone vocab

ularyNone binaryFalse

dtypeclass 'numpyfloat64'

norm'l2' useidfTrue

smoothidfTrue sublin

earthFalse

Convert a collection of raw documents to a matrix of TFIDF features

Equivalent to CountVectorizer followed by TfidfTransformer

Read more in the User Guide

Parameters

input string 'filename' 'file' 'content' If 'filename' the sequence passed as an argument to

fit is expected to be a list of filenames that need reading to fetch the raw content to analyze

If 'file' the sequence items must have a 'read' method filelike object that is called to fetch

the bytes in memory

Otherwise the input is expected to be the sequence strings or bytes items are expected to be

analyzed directly

encoding string 'utf8' by default If bytes or files are given to analyze this encoding is used

to decode

decodeerror 'strict' 'ignore' 'replace' default'strict' Instruction on what to do if a

byte sequence is given to analyze that contains characters not of the given encoding By

default it is 'strict' meaning that a UnicodeDecodeError will be raised Other values are

'ignore' and 'replace'

stripaccents 'ascii' 'unicode' None defaultNone Remove accents and perform other

character normalization during the preprocessing step 'ascii' is a fast method that only

works on characters that have an direct ASCII mapping 'unicode' is a slightly slower

method that works on any characters None default does nothing

Both 'ascii' and 'unicode' use NFKD normalization from unicodedatanormalize

lowercase boolean defaultTrue Convert all characters to lowercase before tokenizing

preprocessor callable or None defaultNone Override the preprocessing string transforma

tion stage while preserving the tokenizing and ngrams generation steps

tokenizer callable or None defaultNone Override the string tokenization step while pre

serving the preprocessing and ngrams generation steps Only applies if analyzer

word

analyzer string 'word' 'char' 'charwb' or callable Whether the feature should be made

of word or character ngrams Option 'charwb' creates character ngrams only from text

inside word boundaries ngrams at the edges of words are padded with space

615sklearnfeatureextraction Feature Extraction 1759

scikitlearn user guide Release 0213

If a callable is passed it is used to extract the sequence of features out of the raw unprocessed input

Changed in version 021

Since v021 if input is filename or file the data is first read from the file and then passed to the given callable analyzer

stopwords string 'english' list or None default None If a string it is passed to checkstoplist and the appropriate stop list is returned 'english' is currently the only supported string value There are several known issues with 'english' and you should consider an alternative see Using stop words

If a list that list is assumed to contain stop words all of which will be removed from the resulting tokens Only applies if analyzer word

If None no stop words will be used maxdf can be set to a value in the range 0 7 10 to automatically detect and filter stop words based on intra corpus document frequency of terms

tokenpattern string Regular expression denoting what constitutes a "token" only used if analyzer word The default regexp selects tokens of 2 or more alphanumeric characters punctuation is completely ignored and always treated as a token separator ngramrange tuple minn maxx default 1 1 The lower and upper boundary of the range of nvalues for different ngrams to be extracted All values of n such that minn n maxx will be used

maxdf float in range 0 10 or int default 10 When building the vocabulary ignore terms that have a document frequency strictly higher than the given threshold corpus specific stop words If float the parameter represents a proportion of documents integer absolute counts This parameter is ignored if vocabulary is not None mindf float in range 0 10 or int default 1 When building the vocabulary ignore terms that have a document frequency strictly lower than the given threshold This value is also called cutoff in the literature If float the parameter represents a proportion of documents integer absolute counts This parameter is ignored if vocabulary is not None maxfeatures int or None default None If not None build a vocabulary that only consider the top maxfeatures ordered by term frequency across the corpus

This parameter is ignored if vocabulary is not None

vocabulary Mapping or iterable optional default None Either a Mapping eg a dict where keys are terms and values are indices in the feature matrix or an iterable over terms If not given a vocabulary is determined from the input documents

binary boolean default False If True all nonzero term counts are set to 1 This does not mean outputs will have only 0 1 values only that the tf term in tfidf is binary Set idf and normalization to False to get 0 1 outputs

dtype type optional default float64 Type of the matrix returned by fittransform or transform

norm 'l1' 'l2' or None optional default 'l2' Each output row will have unit norm either 'l2' Sum of squares of vector elements is 1 The cosine similarity between two vectors is their dot product when l2 norm has been applied 'l1' Sum of absolute values of vector elements is 1 See preprocessingnormalize

useidf boolean default True Enable inversedocumentfrequency reweighting

scikitlearn user guide Release 0213

smoothidf boolean defaultTrue Smooth idf weights by adding one to document frequencies as if an extra document was seen containing every term in the collection exactly once Prevents zero divisions

sublineartf boolean defaultFalse Apply sublinear tf scaling ie replace tf with 1 logtf

Attributes

vocabulary dict A mapping of terms to feature indices

idf array shape nfeatures The inverse document frequency IDF vector only defined if useidf is True

stopwords set Terms that were ignored because they either

- occurred in too many documents maxdf
- occurred in too few documents mindf
- were cut off by feature selection maxfeatures

This is only available if no vocabulary was given

See also

CountVectorizer Transforms text into a sparse matrix of ngram counts

TfidfTransformer Performs the TFIDF transformation from a provided matrix of counts

Notes

Thestopwords attribute can get large and increase the model size when pickling This attribute is provided only for introspection and can be safely removed using delattr or set to None before pickling

Examples

```
from sklearn.feature_extraction.text import TfidfVectorizer
corpus
```

This is the first document

This document is the second document

And this is the third one

Is this the first document

```
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(corpus)
```

```
print(vectorizer.get_feature_names())
# and document first is one second the third this
```

```
print(X.shape)
# 4 9
```

Methods

build_analyzer self Return a callable that handles preprocessing and tokenization

Continued on next page

615sklearn.feature_extraction Feature Extraction 1761

`buildpreprocessor` self Return a function to preprocess the text before tokenization

`buildtokenizer` self Return a function that splits a string into a sequence of tokens

`decode` self doc Decode the input into a string of unicode symbols

`fitself` rawdocuments y Learn vocabulary and idf from training set

`fittransform` self rawdocuments y Learn vocabulary and idf return termdocument matrix

`getfeaturenames` self Array mapping from feature integer indices to feature name

`getparams` self deep Get parameters for this estimator

`getstopwords` self Build or fetch the effective stop words list

`inversetransform` self X Return terms per document with nonzero entries in X

`setparams` self params Set the parameters of this estimator

`transform` self rawdocuments copy Transform documents to documentterm matrix

`init` selfinput'content' encoding'utf8' decodeerror'strict' stripaccentsNone lowercaseTrue preprocessorNone tokenizerNone analyzer'word' stopwordsNone tokenpattern'ubwwb' ngramrange1 1maxdf10 mindf1 maxfeaturesNone vocabularyNone binaryFalse dtypeclass 'numpyfloat64' norm'l2' useidfTrue smoothidfTrue sublineartfFalse

`buildanalyzer` self

Return a callable that handles preprocessing and tokenization

`buildpreprocessor` self

Return a function to preprocess the text before tokenization

`buildtokenizer` self

Return a function that splits a string into a sequence of tokens

`decode` selfdoc

Decode the input into a string of unicode symbols

The decoding strategy depends on the vectorizer parameters

Parameters

doc string The string to decode

`fitself`rawdocuments yNone

Learn vocabulary and idf from training set

Parameters

rawdocuments iterable an iterable which yields either str unicode or file objects

Returns

self TfIdfVectorizer

`fittransform` selfrawdocuments yNone

Learn vocabulary and idf return termdocument matrix

This is equivalent to fit followed by transform but more efficiently implemented

Parameters

rawdocuments iterable an iterable which yields either str unicode or file objects

Returns

Xsparse matrix nsamples nfeatures TfIdfweighted documentterm matrix

1762 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getfeaturenames self

Array mapping from feature integer indices to feature name

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getstopwords self

Build or fetch the effective stop words list

inversetransform selfX

Return terms per document with nonzero entries in X

Parameters

Xarray sparse matrix shape nsamples nfeatures

Returns

Xinv list of arrays len nsamples List of arrays of terms

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfrawdocuments copyTrue

Transform documents to documentterm matrix

Uses the vocabulary and document frequencies df learned by fit or fittransform

Parameters

rawdocuments iterable an iterable which yields either str unicode or file objects

copy boolean default True Whether to copy X and operate on the copy or perform inplace

operations

Returns

Xsparse matrix nsamples nfeatures Tfidfweighted documentterm matrix

Examples using sklearnfeatureextractiontextTfidfVectorizer

•Topic extraction with Nonnegative Matrix Factorization and Latent Dirichlet Allocation

•Biclustering documents with the Spectral Coclustering algorithm

•Column Transformer with Heterogeneous Data Sources

•Clustering text documents using kmeans

615sklearnfeatureextraction Feature Extraction 1763

scikitlearn user guide Release 0213

•Classification of text documents using sparse features

616sklearnfeatureselection Feature Selection

The sklearnfeatureselection module implements feature selection algorithms It currently includes univariate filter selection methods and the recursive feature elimination algorithm

User guide See the Feature selection section for further details

featureselection

GenericUnivariateSelect Univariate feature selector with configurable strategy

featureselectionSelectPercentile Select features according to a percentile of the highest scores

featureselectionSelectKBest scorefunc

kSelect features according to the k highest scores

featureselectionSelectFpr scorefunc al

phaFilter Select the pvalues below alpha based on a FPR test

featureselectionSelectFdr scorefunc al

phaFilter Select the pvalues for an estimated false discovery rate

featureselection

SelectFromModel estimatorMetatransformer for selecting features based on importance weights

featureselectionSelectFwe scorefunc al

phaFilter Select the pvalues corresponding to Familywise error rate

featureselectionRFE estimator Feature ranking with recursive feature elimination

featureselectionRFECV estimator step Feature ranking with recursive feature elimination and crossvalidated selection of the best number of features

featureselection

VarianceThreshold thresholdFeature selector that removes all lowvariance features

6161sklearnfeatureselection GenericUnivariateSelect

class sklearnfeatureselection GenericUnivariateSelect scorefuncfunction

fclassif mode'percentile'

param1e05

Univariate feature selector with configurable strategy

Read more in the User Guide

Parameters

scorefunc callable Function taking two arrays X and y and returning a pair of arrays scores

pvalues For modes 'percentile' or 'kbest' it can return a single array scores

mode 'percentile' 'kbest' 'fpr' 'fdr' 'fwe' Feature selection mode

param float or int depending on the feature selection mode Parameter of the corresponding mode

Attributes

scores arraylike shapenfeatures Scores of features

pvalues arraylike shapenfeatures pvalues of feature scores None if scorefunc

returned scores only

See also

1764 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

fclassif ANOV A Fvalue between labelfeature for classification tasks

mutualinfoclassif Mutual information for a discrete target

chi2 Chisquared stats of nonnegative features for classification tasks

fregression Fvalue between labelfeature for regression tasks

mutualinfo regression Mutual information for a continuous target

SelectPercentile Select features based on percentile of the highest scores

SelectKBest Select features based on the k highest scores

SelectFpr Select features based on a false positive rate test

SelectFdr Select features based on an estimated false discovery rate

SelectFwe Select features based on familywise error rate

Examples

```
from sklearn.datasets import loadbreastcancer
from sklearn.feature_selection import GenericUnivariateSelect chi2
X y loadbreastcancerreturnXy True
Xshape
569 30
transformer GenericUnivariateSelectchi2 kbest param20
Xnew transformerfittransformX y
Xnewshape
569 20
```

Methods

fitself X y Run score function on X y and get the appropriate features

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

getsupport self indices Get a mask or integer index of the features selected

inversetransform self X Reverse the transformation operation

setparams self params Set the parameters of this estimator

transform self X Reduce X to the selected features

init selfscorefuncfunction fclassif at 0x7efe30bb2158 mode'percentile' param1e05

fitselfXy

Run score function on X y and get the appropriate features

Parameters

Xarraylike shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values class labels in classification real numbers in regression

Returns

616sklearn.feature_selection Feature Selection 1765

scikitlearn user guide Release 0213

self object
fittransform selfXyNone fitparams
Fit to data then transform it
Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X
Parameters
Xnumpy array of shape nsamples nfeatures Training set
ynumpy array of shape nsamples Target values
Returns
Xnew numpy array of shape nsamples nfeaturesnew Transformed array
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values
getsupport selfindicesFalse
Get a mask or integer index of the features selected
Parameters
indices boolean default False If True the return value will be an array of integers rather
than a boolean mask
Returns
support array An index that selects the retained features from a feature vector If
indices is False this is a boolean array of shape input features in which an ele
ment is True iff its corresponding feature is selected for retention If indices is True
this is an integer array of shape output features whose values are indices into the input
feature vector
inversetransform selfX
Reverse the transformation operation
Parameters
Xarray of shape nsamples nselectedfeatures The input samples
Returns
Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted
where features would have been removed by transform
setparams selfparams
Set the parameters of this estimator
The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it's possible to update each component
of a nested object
Returns
self

scikitlearn user guide Release 0213

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the selected features

6162sklearnfeatureselection SelectPercentile

classssklearnfeatureselection SelectPercentile scorefuncfunction fclassif percentile10

Select features according to a percentile of the highest scores

Read more in the User Guide

Parameters

scorefunc callable Function taking two arrays X and y and returning a pair of arrays scores pvalues or a single array with scores Default is fclassif see below “See also” The default function only works with classification tasks

percentile int optional default10 Percent of features to keep

Attributes

scores arraylike shapenfeatures Scores of features

pvalues arraylike shapenfeatures pvalues of feature scores None if scorefunc returned only scores

See also

fclassif ANOV A Fvalue between labelfeature for classification tasks

mutualinfoclassif Mutual information for a discrete target

chi2 Chisquared stats of nonnegative features for classification tasks

fregression Fvalue between labelfeature for regression tasks

mutualinfo regression Mutual information for a continuous target

SelectKBest Select features based on the k highest scores

SelectFpr Select features based on a false positive rate test

SelectFdr Select features based on an estimated false discovery rate

SelectFwe Select features based on familywise error rate

GenericUnivariateSelect Univariate feature selector with configurable mode

Notes

Ties between features with equal scores will be broken in an unspecified way

6163sklearnfeatureselection Feature Selection 1767

scikitlearn user guide Release 0213

Examples

```
from sklearn.datasets import load_digits
from sklearn.feature_selection import SelectPercentile chi2
```

```
X y load_digits(return_Xy=True)
```

Xshape

1797 64

```
Xnew SelectPercentile(chi2, percentile=10).fit_transform(X y)
```

Xnewshape

1797 7

Methods

fitself X y Run score function on X y and get the appropriate

features

fit_transform self X y Fit to data then transform it

get_params self deep Get parameters for this estimator

get_support self indices Get a mask or integer index of the features selected

inverse_transform self X Reverse the transformation operation

set_params self params Set the parameters of this estimator

transform self X Reduce X to the selected features

init self score_func function fclassif at 0x7f3c10e93840 percentile=10

fitself X y

Run score function on X y and get the appropriate features

Parameters

Xarraylike shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values class labels in classification real num

bers in regression

Returns

self object

fit_transform self X y None fit_params

Fit to data then transform it

Fits transformer to X and y with optional parameters fit_params and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

get_params self deep True

Get parameters for this estimator

Parameters

1768 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

support array An index that selects the retained features from a feature vector If

indices is False this is a boolean array of shape input features in which an ele

ment is True iff its corresponding feature is selected for retention If indices is True

this is an integer array of shape output features whose values are indices into the input feature vector

inversetransform selfX

Reverse the transformation operation

Parameters

Xarray of shape nsamples nselectedfeatures The input samples

Returns

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted

where features would have been removed by transform

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the selected features

Examples using sklearnfeatureselectionSelectPercentile

•Feature agglomeration vs univariate selection

•Univariate Feature Selection

•SVMAnova SVM (Support Vector Machine) with univariate feature selection

616sklearnfeatureselection Feature Selection 1769

scikitlearn user guide Release 0213

6163sklearnfeatureselection SelectKBest

classsklearnfeatureselection SelectKBest scorefuncfunction fclassif k10

Select features according to the k highest scores

Read more in the User Guide

Parameters

scorefunc callable Function taking two arrays X and y and returning a pair of arrays scores

pvalues or a single array with scores Default is fclassif see below “See also” The

default function only works with classification tasks

kint or “all” optional default10 Number of top features to select The “all” option bypasses

selection for use in a parameter search

Attributes

scores arraylike shapenfeatures Scores of features

pvalues arraylike shapenfeatures pvalues of feature scores None if scorefunc

returned only scores

See also

fclassif ANOV A Fvalue between labelfeature for classification tasks

mutualinfoclassif Mutual information for a discrete target

chi2 Chisquared stats of nonnegative features for classification tasks

fregression Fvalue between labelfeature for regression tasks

mutualinfoforegression Mutual information for a continuous target

SelectPercentile Select features based on percentile of the highest scores

SelectFpr Select features based on a false positive rate test

SelectFdr Select features based on an estimated false discovery rate

SelectFwe Select features based on familywise error rate

GenericUnivariateSelect Univariate feature selector with configurable mode

Notes

Ties between features with equal scores will be broken in an unspecified way

Examples

from sklearndatasets import loadaddigits

from sklearnfeatureselection import SelectKBest chi2

X y loadaddigitsreturnXy True

Xshape

1797 64

Xnew SelectKBestchi2 k20fittransformX y

Xnewshape

1797 20

1770 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Methods

fitself X y Run score function on X y and get the appropriate features

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

getsupport self indices Get a mask or integer index of the features selected

inversetransform self X Reverse the transformation operation

setparams self params Set the parameters of this estimator

transform self X Reduce X to the selected features

init selfscorefuncfunction fclassif at 0x7f3c10e93840 k10

fitselfXy

Run score function on X y and get the appropriate features

Parameters

Xarraylike shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values class labels in classification real numbers in regression

Returns

self object

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

616sklearnfeatureselection Feature Selection 1771

scikitlearn user guide Release 0213

support array An index that selects the retained features from a feature vector If indices is False this is a boolean array of shape input features in which an element is True iff its corresponding feature is selected for retention If indices is True this is an integer array of shape output features whose values are indices into the input feature vector

inversetransform selfX

Reverse the transformation operation

Parameters

Xarray of shape nsamples nselectedfeatures The input samples

Returns

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted where features would have been removed by transform

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the selected features

Examples using sklearnfeatureselectionSelectKBest

- Concatenating multiple feature extraction methods
- Selecting dimensionality reduction with Pipeline and GridSearchCV
- Pipeline Anova SVM (Support Vector Machine)
- Classification of text documents using sparse features

6164sklearnfeatureselection SelectFpr

classsklearnfeatureselection SelectFpr scorefuncfunction fclassif alpha005

Filter Select the pvalues below alpha based on a FPR test

FPR test stands for False Positive Rate test It controls the total amount of false detections

Read more in the User Guide

Parameters

1772 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

scorefunc callable Function taking two arrays X and y and returning a pair of arrays scores
pvalues Default is fclassif see below “See also” The default function only works with
classification tasks

alpha float optional The highest pvalue for features to be kept

Attributes

scores arraylike shapenfeatures Scores of features
pvalues arraylike shapenfeatures pvalues of feature scores

See also

fclassif ANOV A Fvalue between labelfeature for classification tasks

chi2 Chisquared stats of nonnegative features for classification tasks

mutualinfoclassif

fregression Fvalue between labelfeature for regression tasks

mutualinfoforegression Mutual information between features and the target

SelectPercentile Select features based on percentile of the highest scores

SelectKBest Select features based on the k highest scores

SelectFdr Select features based on an estimated false discovery rate

SelectFwe Select features based on familywise error rate

GenericUnivariateSelect Univariate feature selector with configurable mode

Examples

```
from sklearndatasets import loadbreastcancer
from sklearnfeatureselection import SelectFpr chi2
X y loadbreastcancerreturnXy True
Xshape
569 30
Xnew SelectFprchi2 alpha001fittransformX y
Xnewshape
569 16
```

Methods

fitself X y Run score function on X y and get the appropriate
features

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

getsupport self indices Get a mask or integer index of the features selected

inversetransform self X Reverse the transformation operation

setparams self params Set the parameters of this estimator

transform self X Reduce X to the selected features

init selfscorefuncfunction fclassif at 0x7efe30bb2158 alpha005

616sklearnfeatureselection Feature Selection 1773

scikitlearn user guide Release 0213

fitselfXy

Run score function on X y and get the appropriate features

Parameters

Xarraylike shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values class labels in classification real num

bers in regression

Returns

self object

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

support array An index that selects the retained features from a feature vector If

indices is False this is a boolean array of shape input features in which an ele

ment is True iff its corresponding feature is selected for retention If indices is True

this is an integer array of shape output features whose values are indices into the input feature vector

inversetransform selfX

Reverse the transformation operation

Parameters

Xarray of shape nsamples nselectedfeatures The input samples

Returns

1774 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted
where features would have been removed by transform

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it’s possible to update each component
of a nested object

Returns

self

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the se
lected features

6165sklearnfeatureselection SelectFdr

classssklearnfeatureselection SelectFdr scorefuncfunction fclassif alpha005

Filter Select the pvalues for an estimated false discovery rate

This uses the BenjaminiHochberg procedure alpha is an upper bound on the expected false discovery rate

Read more in the User Guide

Parameters

scorefunc callable Function taking two arrays X and y and returning a pair of arrays scores

pvalues Default is fclassif see below “See also” The default function only works with
classification tasks

alpha float optional The highest uncorrected pvalue for features to keep

Attributes

scores arraylike shapenfeatures Scores of features

pvalues arraylike shapenfeatures pvalues of feature scores

See also

fclassif ANOV A Fvalue between labelfeature for classification tasks

mutualinfoclassif Mutual information for a discrete target

chi2 Chisquared stats of nonnegative features for classification tasks

fregression Fvalue between labelfeature for regression tasks

mutualinfoforegression Mutual information for a contnuous target

SelectPercentile Select features based on percentile of the highest scores

SelectKBest Select features based on the k highest scores

SelectFpr Select features based on a false positive rate test

616sklearnfeatureselection Feature Selection 1775

scikitlearn user guide Release 0213

SelectFwe Select features based on familywise error rate

GenericUnivariateSelect Univariate feature selector with configurable mode

References

https://en.wikipedia.org/wiki/False_discovery_rate

Examples

```
from sklearn.datasets import loadbreastcancer
from sklearn.feature_selection import SelectFdr_chi2
X, y = loadbreastcancer(return_Xy=True)
X.shape
(569, 30)
X_new = SelectFdr_chi2(alpha=0.01).fit_transform(X, y)
X_new.shape
(569, 16)
```

Methods

fitself X y Run score function on X y and get the appropriate features

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

getsupport self indices Get a mask or integer index of the features selected

inversetransform self X Reverse the transformation operation

setparams self params Set the parameters of this estimator

transform self X Reduce X to the selected features

init selfscorefuncfunction fclassif at 0x7efe30bb2158 alpha005

fitselfXy

Run score function on X y and get the appropriate features

Parameters

Xarraylike shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values class labels in classification real numbers in regression

Returns

self object

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

1776 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

support array An index that selects the retained features from a feature vector If indices is False this is a boolean array of shape input features in which an element is True iff its corresponding feature is selected for retention If indices is True this is an integer array of shape output features whose values are indices into the input feature vector

inversetransform selfX

Reverse the transformation operation

Parameters

Xarray of shape nsamples nselectedfeatures The input samples

Returns

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted where features would have been removed by transform

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

616sklearnfeatureselection Feature Selection 1777

scikitlearn user guide Release 0213

Xr array of shape nsamples nselectedfeatures The input samples with only the selected features

6166sklearnfeatureselection SelectFromModel

classsklearnfeatureselection SelectFromModel estimator thresholdNone prefitFalse normorder1 maxfeaturesNone

Metatransformer for selecting features based on importance weights

New in version 017

Parameters

estimator object The base estimator from which the transformer is built This can be both a fitted ifprefit is set to True or a nonfitted estimator The estimator must have either a featureimportances orcoef attribute after fitting

threshold string float optional default None The threshold value to use for feature selection

Features whose importance is greater or equal are kept while the others are discarded If

“median” resp “mean” then the threshold value is the median resp the mean of the

feature importances A scaling factor eg “125mean” may also be used If None and

if the estimator has a parameter penalty set to l1 either explicitly or implicitly eg Lasso

the threshold used is 1e5 Otherwise “mean” is used by default

prefit bool default False Whether a prefit model is expected to be passed into the constructor directly or not If True transform must be called directly and SelectFromModel can

not be used with crossvalscore GridSearchCV and similar utilities that clone

the estimator Otherwise train the model using fit and thentransform to do feature

selection

normorder nonzero int inf inf default 1 Order of the norm used to filter the vectors of coefficients below threshold in the case where the coef attribute of the estimator is of dimension 2

maxfeatures int or None optional The maximum number of features selected scoring above threshold To disable threshold and only select based on maxfeatures set

thresholdnpinf

New in version 020

Attributes

estimator an estimator The base estimator from which the transformer is built This is stored only when a nonfitted estimator is passed to the SelectFromModel ie when prefit is

False

threshold float The threshold value used for feature selection

Methods

fitself X y Fit the SelectFromModel metatransformer

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

getsupport self indices Get a mask or integer index of the features selected

inversetransform self X Reverse the transformation operation

partialfit self X y Fit the SelectFromModel metatransformer only once

Continued on next page

1778 Chapter 6 API (Application Programming Interface) Reference

setparams self params Set the parameters of this estimator

transform self X Reduce X to the selected features

init selfestimator thresholdNone prefitFalse normorder1 maxfeaturesNone

fitselfXyNone fitparams

Fit the SelectFromModel metatransformer

Parameters

Xarraylike of shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values integers that correspond to classes in classification real numbers in regression

fitparams Other estimator specific parameters

Returns

self object

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

support array An index that selects the retained features from a feature vector If

indices is False this is a boolean array of shape input features in which an ele

ment is True iff its corresponding feature is selected for retention If indices is True

this is an integer array of shape output features whose values are indices into the input

feature vector

scikitlearn user guide Release 0213

inversetransform selfX

Reverse the transformation operation

Parameters

Xarray of shape nsamples nselectedfeatures The input samples

Returns

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted

where features would have been removed by transform

partialfit selfXyNone fitparams

Fit the SelectFromModel metatransformer only once

Parameters

Xarraylike of shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values integers that correspond to classes in

classification real numbers in regression

fitparams Other estimator specific parameters

Returns

self object

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the se

lected features

Examples using sklearnfeatureselectionSelectFromModel

•Feature selection using SelectFromModel and LassoCV

•Classification of text documents using sparse features

6167sklearnfeatureselection SelectFwe

classsklearnfeatureselection SelectFwe scorefuncfunction fclassif alpha005

Filter Select the pvalues corresponding to Familywise error rate

Read more in the User Guide

1780 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

scorefunc callable Function taking two arrays X and y and returning a pair of arrays scores

pvalues Default is fclassif see below “See also” The default function only works with

classification tasks

alpha float optional The highest uncorrected pvalue for features to keep

Attributes

scores arraylike shapenfeatures Scores of features

pvalues arraylike shapenfeatures pvalues of feature scores

See also

fclassif ANOV A Fvalue between labelfeature for classification tasks

chi2 Chisquared stats of nonnegative features for classification tasks

fregression Fvalue between labelfeature for regression tasks

SelectPercentile Select features based on percentile of the highest scores

SelectKBest Select features based on the k highest scores

SelectFpr Select features based on a false positive rate test

SelectFdr Select features based on an estimated false discovery rate

GenericUnivariateSelect Univariate feature selector with configurable mode

Examples

from sklearndatasets import loadbreastcancer

from sklearnfeatureselection import SelectFwe chi2

X y loadbreastcancerreturnXy True

Xshape

569 30

Xnew SelectFwechi2 alpha001fittransformX y

Xnewshape

569 15

Methods

fitself X y Run score function on X y and get the appropriate

features

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

getsupport self indices Get a mask or integer index of the features selected

inversetransform self X Reverse the transformation operation

setparams self params Set the parameters of this estimator

transform self X Reduce X to the selected features

init selfscorefuncfunction fclassif at 0x7efe30bb2158 alpha005

fitselfXy

616sklearnfeatureselection Feature Selection 1781

scikitlearn user guide Release 0213

Run score function on X y and get the appropriate features

Parameters

Xarraylike shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values class labels in classification real numbers in regression

Returns

self object

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

support array An index that selects the retained features from a feature vector If

indices is False this is a boolean array of shape input features in which an element

is True iff its corresponding feature is selected for retention If indices is True

this is an integer array of shape output features whose values are indices into the input feature vector

inversetransform selfX

Reverse the transformation operation

Parameters

Xarray of shape nsamples nselectedfeatures The input samples

Returns

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted

where features would have been removed by transform

1782 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the selected features

6168sklearnfeatureselection RFE

classsklearnfeatureselection RFEestimator nfeaturestoselectNone step1 verbose0

Feature ranking with recursive feature elimination

Given an external estimator that assigns weights to features eg the coefficients of a linear model the goal of recursive feature elimination RFE is to select features by recursively considering smaller and smaller sets of features First the estimator is trained on the initial set of features and the importance of each feature is obtained either through a coef attribute or through a featureimportances attribute Then the least important features are pruned from current set of features That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached

Read more in the User Guide

Parameters

estimator object A supervised learning estimator with a fit method that provides information about feature importance either through a coef attribute or through a featureimportances attribute

nfeaturestoselect int or None defaultNone The number of features to select If None half of the features are selected

step int or float optional default1 If greater than or equal to 1 then step corresponds to the integer number of features to remove at each iteration If within 00 10 then step corresponds to the percentage rounded down of features to remove at each iteration

verbose int default0 Controls verbosity of output

Attributes

nfeatures int The number of selected features

support array of shape nfeatures The mask of selected features

ranking array of shape nfeatures The feature ranking such that rankingi corresponds to the ranking position of the ith feature Selected ie estimated best features are assigned rank 1

estimator object The external estimator fit on the reduced dataset

616sklearnfeatureselection Feature Selection 1783

scikitlearn user guide Release 0213

See also

RFE CV Recursive feature elimination with builtin crossvalidated selection of the best number of features

References

Re310f679c81e1

Examples

The following example shows how to retrieve the 5 right informative features in the Friedman 1 dataset

```
from sklearn.datasets import makefriedman1
```

```
from sklearn.feature_selection import RFE
```

```
from sklearn.svm import SVR (Support Vector Regression)
```

```
X, y = makefriedman1(n_samples=50, n_features=10, random_state=0)
```

```
estimator = SVR(kernel='linear')
```

```
selector = RFE(estimator, 5, step=1)
```

```
selector.fit(X, y)
```

```
selector.support
```

```
array([ True,  True,  True,  True,  True, False, False, False, False,
```

```
       False])
```

```
selector.ranking
```

```
array([1, 1, 1, 1, 1, 6, 4, 3, 2, 5])
```

Methods

decisionfunction self X Compute the decision function of X

fit self X y Fit the RFE model and then the underlying estimator on

the selected features

fit_transform self X y Fit to data then transform it

get_params self deep Get parameters for this estimator

get_support self indices Get a mask or integer index of the features selected

inverse_transform self X Reverse the transformation operation

predict self X Reduce X to the selected features and then predict using

the underlying estimator

predict_log_proba self X Predict class logprobabilities for X

predict_proba self X Predict class probabilities for X

score self X y Reduce X to the selected features and then return the

score of the underlying estimator

set_params self params Set the parameters of this estimator

transform self X Reduce X to the selected features

__init__ self estimator n_features_to_select=None step=1 verbose=0

decisionfunction self X

Compute the decision function of X

Parameters

X array-like or sparse matrix shape (n_samples, n_features) The input samples Inter

nally it will be converted to dtype np.float32 and if a sparse matrix is provided to

1784 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

a sparsecsrmatrix

Returns

score array shape nsamples nclasses or nsamples The decision function of the input samples The order of the classes corresponds to that in the attribute classes Regression and binary classification produce an array of shape nsamples

fitselfXy

Fit the RFE model and then the underlying estimator on the selected features

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The training input samples

yarraylike shape nsamples The target values

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

support array An index that selects the retained features from a feature vector If

indices is False this is a boolean array of shape input features in which an element is True iff its corresponding feature is selected for retention If indices is True

this is an integer array of shape output features whose values are indices into the input feature vector

inversetransform selfX

Reverse the transformation operation

Parameters

616sklearnfeatureselection Feature Selection 1785

scikitlearn user guide Release 0213

Xarray of shape nsamples nselectedfeatures The input samples

Returns

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted

where features would have been removed by transform

predictselfX

Reduce X to the selected features and then predict using the underlying estimator

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

yarray of shape nsamples The predicted target values

predictlogproba selfX

Predict class logprobabilities for X

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

parray of shape nsamples nclasses The class logprobabilities of the input samples

The order of the classes corresponds to that in the attribute classes

predictproba selfX

Predict class probabilities for X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Inter

nally it will be converted to dtype npfloat32 and if a sparse matrix is provided to

a sparsecsrmatrix

Returns

parray of shape nsamples nclasses The class probabilities of the input samples The

order of the classes corresponds to that in the attribute classes

scoreselfXy

Reduce X to the selected features and then return the score of the underlying estimator

Parameters

Xarray of shape nsamples nfeatures The input samples

yarray of shape nsamples The target values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

1786 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the selected features

Examples using sklearnfeatureselectionRFE

•Recursive feature elimination

6169sklearnfeatureselection RFECV

classsklearnfeatureselection RFECVestimator step1 minfeaturestoselect1 cv'warn'

scoringNone verbose0 njobsNone

Feature ranking with recursive feature elimination and crossvalidated selection of the best number of features

See glossary entry for crossvalidation estimator

Read more in the User Guide

Parameters

estimator object A supervised learning estimator with a fit method that provides information about feature importance either through a coef attribute or through a featureimportances attribute

step int or float optional default1 If greater than or equal to 1 then step corresponds to the integer number of features to remove at each iteration If within 00 10 then step corresponds to the percentage rounded down of features to remove at each iteration Note that the last iteration may remove fewer than step features in order to reach minfeaturestoselect

minfeaturestoselect int default1 The minimum number of features to be selected

This number of features will always be scored even if the difference between the original feature count and minfeaturestoselect isn't divisible by step

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

• None to use the default 3fold crossvalidation

• integer to specify the number of folds

•CV (Cross-Validation) splitter

• An iterable yielding train test splits as arrays of indices

For integerNone inputs if yis binary or multiclass sklearnmodelselection

StratifiedKFold is used If the estimator is a classifier or if yis neither binary nor

multiclass sklearnmodelselectionKFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value of None will change from 3fold to 5fold in v022

616sklearnfeatureselection Feature Selection 1787

scikitlearn user guide Release 0213

scoring string callable or None optional defaultNone A string see model evaluation doc
umentation or a scorer callable object function with signature scorere estimator
X y

verbose int default0 Controls verbosity of output

njobs int or None optional defaultNone Number of cores to run in parallel while fitting

across folds None means 1 unless in a joblibparallelbackend context1
means using all processors See Glossary for more details

Attributes

nfeatures int The number of selected features with crossvalidation

support array of shape nfeatures The mask of selected features

ranking array of shape nfeatures The feature ranking such that rankingi corre
sponds to the ranking position of the ith feature Selected ie estimated best features are
assigned rank 1

gridscores array of shape nsubsetsoffeatures The crossvalidation scores such that
gridscoresi corresponds to the CV (Cross-Validation) score of the ith subset of features

estimator object The external estimator fit on the reduced dataset

See also

RFE Recursive feature elimination

Notes

The size of gridscores is equal to ceil(nfeatures / minfeaturestoselect
step / 1 where step is the number of features removed at each iteration

References

R6f4d61ceb4111

Examples

The following example shows how to retrieve the apriori not known 5 informative features in the Friedman 1
dataset

```
from sklearn.datasets import makefriedman1
from sklearn.featureselection import RFECV
from sklearn.svm import SVR (Support Vector Regression)
X y = makefriedman1(nsamples=50, nfeatures=10, randomstate=0)
estimator = SVR(kernel=linear)
selector = RFECV(estimator, step=1, cv=5)
selector.fit(X, y)
selector.support
array [True True True True True False False False False
False]
selector.ranking
array [1 1 1 1 1 6 4 3 2 5]
```

1788 Chapter 6 API (Application Programming Interface) Reference

Methods

decisionfunction self X Compute the decision function of X
fitself X y groups Fit the RFE model and automatically tune the number of selected features
fittransform self X y Fit to data then transform it
getparams self deep Get parameters for this estimator
getsupport self indices Get a mask or integer index of the features selected
inversetransform self X Reverse the transformation operation
predict self X Reduce X to the selected features and then predict using the underlying estimator
predictlogproba self X Predict class logprobabilities for X
predictproba self X Predict class probabilities for X
score self X y Reduce X to the selected features and then return the score of the underlying estimator
setparams self params Set the parameters of this estimator
transform self X Reduce X to the selected features
init selfestimator step1 minfeaturestoselect1 cv'warn' scoringNone verbose0
njobsNone

decisionfunction selfX
Compute the decision function of X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

Returns

score array shape nsamples nclasses or nsamples The decision function of the input samples The order of the classes corresponds to that in the attribute classes Regression and binary classification produce an array of shape nsamples
fitselfXygroupsNone

Fit the RFE model and automatically tune the number of selected features

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the total number of features
yarraylike shape nsamples Target values integers for classification real numbers for regression
groups arraylike shape nsamples optional Group labels for the samples used while splitting the dataset into train/test set Only used in conjunction with a "Group" cvinstance
egGroupKFold

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

616sklearnfeatureselection Feature Selection 1789

scikitlearn user guide Release 0213

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

support array An index that selects the retained features from a feature vector If indices is False this is a boolean array of shape input features in which an element is True iff its corresponding feature is selected for retention If indices is True this is an integer array of shape output features whose values are indices into the input feature vector

inversetransform selfX

Reverse the transformation operation

Parameters

Xarray of shape nsamples nselectedfeatures The input samples

Returns

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted where features would have been removed by transform

predictselfX

Reduce X to the selected features and then predict using the underlying estimator

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

yarray of shape nsamples The predicted target values

predictlogproba selfX

Predict class logprobabilities for X

Parameters

Xarray of shape nsamples nfeatures The input samples

1790 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

pararray of shape nsamples nclasses The class logprobabilities of the input samples
The order of the classes corresponds to that in the attribute classes

predictproba selfX

Predict class probabilities for X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Inter
nally it will be converted to dtype npfloat32 and if a sparse matrix is provided to
a sparsecsr matrix

Returns

pararray of shape nsamples nclasses The class probabilities of the input samples The
order of the classes corresponds to that in the attribute classes

scoreselfXy

Reduce X to the selected features and then return the score of the underlying estimator

Parameters

Xarray of shape nsamples nfeatures The input samples

yarray of shape nsamples The target values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it's possible to update each component
of a nested object

Returns

self

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the se
lected features

Examples using sklearnfeatureselectionRFECV

•Recursive feature elimination with crossvalidation

61610sklearnfeatureselection VarianceThreshold

classsklearnfeatureselection VarianceThreshold threshold00

Feature selector that removes all lowvariance features

616sklearnfeatureselection Feature Selection 1791

scikitlearn user guide Release 0213

This feature selection algorithm looks only at the features X not the desired outputs y and can thus be used for unsupervised learning

Read more in the User Guide

Parameters

threshold float optional Features with a trainingset variance lower than this threshold will be removed The default is to keep all features with nonzero variance ie remove the features that have the same value in all samples

Attributes

variances array shape nfeatures Variances of individual features

Examples

The following dataset has integer features two of which are the same in every sample These are removed with the default setting for threshold

X 0 2 0 3 0 1 4 3 0 1 1 3

selector VarianceThreshold

selectorfittransformX

array2 0

1 4

1 1

Methods

fitself X y Learn empirical variances from X

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

getsupport self indices Get a mask or integer index of the features selected

inversetransform self X Reverse the transformation operation

setparams self params Set the parameters of this estimator

transform self X Reduce X to the selected features

init selfthreshold00

fitselfXyNone

Learn empirical variances from X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Sample vectors from which to compute variances

yany Ignored This parameter exists only for compatibility with sklearnpipelinePipeline

Returns

self

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

1792 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getsupport selfindicesFalse

Get a mask or integer index of the features selected

Parameters

indices boolean default False If True the return value will be an array of integers rather than a boolean mask

Returns

support array An index that selects the retained features from a feature vector If

indices is False this is a boolean array of shape input features in which an ele

ment is True iff its corresponding feature is selected for retention If indices is True

this is an integer array of shape output features whose values are indices into the input feature vector

inversetransform selfX

Reverse the transformation operation

Parameters

Xarray of shape nsamples nselectedfeatures The input samples

Returns

Xr array of shape nsamples noriginalfeatures Xwith columns of zeros inserted

where features would have been removed by transform

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Reduce X to the selected features

Parameters

Xarray of shape nsamples nfeatures The input samples

616sklearnfeatureselection Feature Selection 1793

scikitlearn user guide Release 0213

Returns

Xr array of shape nsamples nselectedfeatures The input samples with only the selected features

featureselectionchi2 X y Compute chisquared stats between each nonnegative feature and class

featureselectionfclassif X y Compute the ANOV A Fvalue for the provided sample

featureselectionfregression X y cen

terUnivariate linear regression tests

featureselection

mutualinfoclassif X yEstimate mutual information for a discrete target variable

featureselection

mutualinfofregression X yEstimate mutual information for a continuous target variable

61611sklearnfeatureselection chi2

sklearnfeatureselection chi2Xy

Compute chisquared stats between each nonnegative feature and class

This score can be used to select the nfeatures features with the highest values for the test chisquared statistic from X which must contain only nonnegative features such as booleans or frequencies eg term counts in document classification relative to the classes

Recall that the chisquare test measures dependence between stochastic variables so using this function “weeds out” the features that are the most likely to be independent of class and therefore irrelevant for classification

Read more in the User Guide

Parameters

Xarraylike sparse matrix shape nsamples nfeaturesin Sample vectors

yarraylike shape nsamples Target vector class labels

Returns

chi2 array shape nfeatures chi2 statistics of each feature

pval array shape nfeatures pvalues of each feature

See also

fclassif ANOV A Fvalue between labelfeature for classification tasks

fregression Fvalue between labelfeature for regression tasks

Notes

Complexity of this algorithm is Onclasses nfeatures

Examples using sklearnfeatureselectionchi2

- Selecting dimensionality reduction with Pipeline and GridSearchCV
- SVMAnova SVM (Support Vector Machine) with univariate feature selection
- Classification of text documents using sparse features

1794 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
61612sklearnfeatureselection fclassif
sklearnfeatureselection fclassif Xy
Compute the ANOV A Fvalue for the provided sample
Read more in the User Guide
Parameters
Xarraylike sparse matrix shape nsamples nfeatures The set of regressors that will
be tested sequentially
yarray of shapensamples The data matrix
Returns
Farray shape nfeatures The set of F values
pval array shape nfeatures The set of pvalues
See also
chi2 Chisquared stats of nonnegative features for classification tasks
fregression Fvalue between labelfeature for regression tasks
Examples using sklearnfeatureselectionfclassif
•Univariate Feature Selection
61613sklearnfeatureselection fregression
sklearnfeatureselection fregression XycenterTrue
Univariate linear regression tests
Linear model for testing the individual effect of each of many regressors This is a scoring function to be used
in a feature selection procedure not a free standing feature selection procedure
This is done in 2 steps
1 The correlation between each regressor and the target is computed that is $X_i - \text{mean}X_i - y$
 $\text{mean}y - \text{std}X_i - \text{std}y$
2 It is converted to an F score then to a pvalue
For more on usage see the User Guide
Parameters
Xarraylike sparse matrix shape nsamples nfeatures The set of regressors that will
be tested sequentially
yarray of shapensamples The data matrix
center True bool If true X and y will be centered
Returns
Farray shapenfeatures F values of features
pval array shapenfeatures pvalues of Fscores
See also
616sklearnfeatureselection Feature Selection 1795

scikitlearn user guide Release 0213

mutualinfo: regression Mutual information for a continuous target

fclassif: ANOVA F-value between label/feature for classification tasks

chi2: Chi-squared stats of nonnegative features for classification tasks

SelectKBest: Select features based on the k highest scores

SelectFpr: Select features based on a false positive rate test

SelectFdr: Select features based on an estimated false discovery rate

SelectFwe: Select features based on familywise error rate

SelectPercentile: Select features based on percentile of the highest scores

Examples using sklearn.feature_selection.regression

- Feature agglomeration vs univariate selection

- Comparison of F-test and mutual information

- Pipeline: ANOVA SVM (Support Vector Machine)

61614 sklearn.feature_selection.mutual_info_classif

sklearn.feature_selection.mutual_info_classif X y discrete features 'auto'

n_neighbors=3 copy=True random_state=None

domstate=None

Estimate mutual information for a discrete target variable

Mutual information MI between two random variables is a nonnegative value which measures the dependency between the variables. It is equal to zero if and only if two random variables are independent and higher values mean higher dependency.

The function relies on nonparametric methods based on entropy estimation from k-nearest neighbors distances as described in [2] and [3]. Both methods are based on the idea originally proposed in [4]. It can be used for univariate features selection. Read more in the User Guide.

Parameters

X: array-like or sparse matrix shape (n_samples, n_features) Feature matrix

y: array-like shape (n_samples) Target vector

discrete_features: 'auto' bool array-like default 'auto' If bool then determines whether to consider all features discrete or continuous. If array then it should be either a boolean mask with shape (n_features) or array with indices of discrete features. If 'auto' it is assigned to False for dense X and to True for sparse X.

n_neighbors: int default 3 Number of neighbors to use for MI estimation for continuous variables. See [2] and [3]. Higher values reduce variance of the estimation but could introduce a bias.

copy: bool default True Whether to make a copy of the given data. If set to False the initial data will be overwritten.

random_state: int RandomState instance or None optional default None The seed of the pseudo random number generator for adding small noise to continuous variables in order to remove repeated values. If int random_state is the seed used by the random number generator.

1796 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Returns

mindarray shape nfeatures Estimated mutual information between each feature and the target

Notes

1 The term “discrete features” is used instead of naming them “categorical” because it describes the essence more accurately For example pixel intensities of an image are discrete features but hardly categorical and you will get better results if mark them as such Also note that treating a continuous variable as discrete and vice versa will usually give incorrect results so be attentive about that

2 True mutual information can’t be negative If its estimate turns out to be negative it is replaced by zero

References

1234

61615sklearnfeatureselection mutualinfo regression

sklearnfeatureselection mutualinfo regression Xy discretefeatures’auto’

nneighbors3 copyTrue ran

domstateNone

Estimate mutual information for a continuous target variable

Mutual information MI 1between two random variables is a nonnegative value which measures the dependency between the variables It is equal to zero if and only if two random variables are independent and higher values mean higher dependency

The function relies on nonparametric methods based on entropy estimation from knearest neighbors distances as described in 2and3 Both methods are based on the idea originally proposed in 4

It can be used for univariate features selection read more in the User Guide

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Feature matrix

yarraylike shape nsamples Target vector

discretefeatures ‘auto’ bool arraylike default ‘auto’ If bool then determines whether to consider all features discrete or continuous If array then it should be either a boolean mask with shape nfeatures or array with indices of discrete features If ‘auto’ it is assigned to False for dense Xand to True for sparse X

nneighbors int default 3 Number of neighbors to use for MI estimation for continuous variables see 2and3 Higher values reduce variance of the estimation but could introduce a bias

copy bool default True Whether to make a copy of the given data If set to False the initial data will be overwritten

randomstate int RandomState instance or None optional default None The seed of the pseudo random number generator for adding small noise to continuous variables in order to remove repeated values If int randomstate is the seed used by the random number 616sklearnfeatureselection Feature Selection 1797

scikitlearn user guide Release 0213

generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Returns

mindarray shape nfeatures Estimated mutual information between each feature and the target

Notes

1 The term “discrete features” is used instead of naming them “categorical” because it describes the essence more accurately For example pixel intensities of an image are discrete features but hardly categorical and you will get better results if mark them as such Also note that treating a continuous variable as discrete and vice versa will usually give incorrect results so be attentive about that

2 True mutual information can’t be negative If its estimate turns out to be negative it is replaced by zero

References

1234

Examples using sklearnfeatureselectionmutualinfo regression

•Comparison of Ftest and mutual information

617sklearn gaussianprocess Gaussian Processes

The sklearn gaussianprocess module implements Gaussian Process based regression and classification

User guide See the Gaussian Processes section for further details

gaussianprocess GaussianProcessClassifier Gaussian process classification GPC based on Laplace approximation

gaussianprocess GaussianProcessRegressor Gaussian process regression GPR

6171 sklearn gaussianprocess GaussianProcessClassifier

class sklearn gaussianprocess GaussianProcessClassifier kernelNone opti

mizer’fminlbfgsb’

nrestartsoptimizer0

maxiterpredict100

warmstartFalse

copyXtrainTrue

randomstateNone

multiclass’onevsrest’

njobsNone

Gaussian process classification GPC based on Laplace approximation

The implementation is based on Algorithm 31 32 and 51 of Gaussian Processes for Machine Learning

GPML by Rasmussen and Williams

1798 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Internally the Laplace approximation is used for approximating the nonGaussian posterior by a Gaussian. Currently the implementation is restricted to using the logistic link function. For multiclass classification several binary oneversus rest classifiers are fitted. Note that this class thus does not implement a true multi-class Laplace approximation.

Parameters

kernel kernel object The kernel specifying the covariance function of the GP. If None is passed the kernel “10 RBF10” is used as default. Note that the kernel’s hyperparameters are optimized during fitting.

optimizer string or callable optional default “fminlbfgsb” Can either be one of the internally supported optimizers for optimizing the kernel’s parameters specified by a string or an externally defined optimizer passed as a callable. If a callable is passed it must have the signature

defoptimizerobjfunc initialtheta bounds

objfunc is the objective function to be maximized which takes the hyperparameters theta as parameter and an optional flag evalgradient which determines if the gradient is returned additionally to the function value.

initialtheta the initial value for theta which can be used by local optimizers.

bounds the bounds on the values of theta.

Returned are the best found hyperparameters theta and the corresponding value of the target function.

returnthetaopt funcmin

Per default the ‘fminlbfgsb’ algorithm from scipy.optimize is used. If None is passed the kernel’s parameters are kept fixed. Available internal optimizers are

fminlbfgsb

nrestartsoptimizer int optional default 0 The number of restarts of the optimizer for finding the kernel’s parameters which maximize the logmarginal likelihood. The first run of the optimizer is performed from the kernel’s initial parameters, the remaining ones if any from thetas sampled loguniform randomly from the space of allowed thetavalues. If greater than 0 all bounds must be finite. Note that nrestartsoptimizer0 implies that one run is performed.

maxiterpredict int optional default 100 The maximum number of iterations in Newton’s method for approximating the posterior during predict. Smaller values will reduce computation time at the cost of worse results.

warmstart bool optional default False If warmstarts are enabled the solution of the last Newton iteration on the Laplace approximation of the posterior mode is used as initialization for the next call of posteriormode. This can speed up convergence when posteriormode is called several times on similar problems as in hyperparameter optimization. See the Glossary.

copyXtrain bool optional default True If True a persistent copy of the training data is stored in the object. Otherwise just a reference to the training data is stored which might cause predictions to change if the data is modified externally.

randomstate int RandomState instance or None optional default None The generator used to initialize the centers. If int randomstate is the seed used by the random number generator.

617sklearn.gaussianprocess Gaussian Processes 1799

scikitlearn user guide Release 0213

generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom
multiclass string default Specifies how multiclass classification problems are handled
Supported are “onevsrest” and “onevsone” In “onevsrest” one binary Gaussian process classifier is fitted for each class which is trained to separate this class from the rest
In “onevsone” one binary Gaussian process classifier is fitted for each pair of classes which is trained to separate these two classes The predictions of these binary predictors are combined into multiclass predictions Note that “onevsone” does not support predicting probability estimates

njobs int or None optional defaultNone The number of jobs to use for the computation
None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

Attributes
kernel kernel object The kernel used for prediction In case of binary classification the structure of the kernel is the same as the one passed as parameter but with optimized hy perparameters In case of multiclass classification a CompoundKernel is returned which consists of the different kernels used in the oneversusrest classifiers
logmarginallikelihoodvalue float The logmarginallikelihood of selfkernel
theta

classes arraylike shape nclasses Unique class labels
nclasses int The number of classes in the training data

Examples
from sklearndatasets import loadiris
from sklearngaussianprocess import GaussianProcessClassifier
from sklearngaussianprocessskernels import RBF (Radial Basis Function)

X y loadirisreturnXy True
kernel 10 RBF10
gpc GaussianProcessClassifierkernelkernel
randomstate0fitX y
gpcscoreX y
09866
gpcpredictprobaX2
array083548752 003228706 013222543
079064206 006525643 014410151

New in version 018

Methods
fitself X y Fit Gaussian process classification model
getparams self deep Get parameters for this estimator
logmarginallikelihood self theta Returns logmarginal likelihood of theta for training data
predict self X Perform classification on an array of test vectors X
predictproba self X Return probability estimates for the test vector X

Continued on next page

1800 Chapter 6 API (Application Programming Interface) Reference

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init self kernelNone optimizer'fminlbfgsb' nrestartsoptimizer0

maxiterpredict100 warmstartFalse copyXtrainTrue randomstateNone

multiclass'onevsrest' njobsNone

fitselfXy

Fit Gaussian process classification model

Parameters

Xarraylike shape nsamples nfeatures Training data

yarraylike shape nsamples Target values must be binary

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

logmarginallikelihood selfthetaNone evalgradientFalse

Returns logmarginal likelihood of theta for training data

In the case of multiclass classification the mean logmarginal likelihood of the oneversusrest classifiers are returned

Parameters

theta arraylike shape nkernelparams or none Kernel hyperparameters for which the logmarginal likelihood is evaluated In the case of multiclass classification theta may be the hyperparameters of the compound kernel or of an individual kernel In the latter case all individual kernel get assigned the same theta values If None the precomputed logmarginallikelihood of selfkerneltheta is returned

evalgradient bool default False If True the gradient of the logmarginal likelihood with respect to the kernel hyperparameters at position theta is returned additionally Note that gradient computation is not supported for nonbinary classification If True theta must not be None

Returns

loglikelihood float Logmarginal likelihood of theta for training data

loglikelihoodgradient array shape nkernelparams optional Gradient of the log marginal likelihood with respect to the kernel hyperparameters at position theta Only returned when evalgradient is True

predictselfX

Perform classification on an array of test vectors X

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carray shape nsamples Predicted target values for X values are from classes

predictproba selfX

Return probability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carraylike shape nsamples nclasses Returns the probability of the samples for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearngaussianprocessGaussianProcessClassifier

- Plot classification probability
 - Classifier comparison
 - Illustration of Gaussian process classification GPC on the XOR dataset
 - Gaussian process classification GPC on iris dataset
 - Isoprobability lines for Gaussian Processes classification GPC
 - Probabilistic predictions with Gaussian process classification GPC
- 1802 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

```
6172sklearnprocess GaussianProcessRegressor
classsklearnprocess GaussianProcessRegressor kernelNone alpha1e10
optimizer'fminlbfgsb'
nrestartoptimizer0
normalizeyFalse
copyXtrainTrue ran
domstateNone
```

Gaussian process regression GPR

The implementation is based on Algorithm 21 of Gaussian Processes for Machine Learning GPML by Ras
mussen and Williams

In addition to standard scikitlearn estimator API (Application Programming Interface) GaussianProcessRegressor

- allows prediction without prior fitting based on the GP prior
- provides an additional method sampleX which evaluates samples drawn from the GPR prior or pos
terior at given inputs
- exposes a method logmarginallikelihoodtheta which can be used externally for other ways of selecting
hyperparameters eg via Markov chain Monte Carlo

Read more in the User Guide

New in version 018

Parameters

kernel kernel object The kernel specifying the covariance function of the GP If None is
passed the kernel “10 RBF10” is used as default Note that the kernel’s hyperpa
rameters are optimized during fitting
alpha float or arraylike optional default 1e10 Value added to the diagonal of the kernel
matrix during fitting Larger values correspond to increased noise level in the observations
This can also prevent a potential numerical issue during fitting by ensuring that the calcu
lated values form a positive definite matrix If an array is passed it must have the same
number of entries as the data used for fitting and is used as datapointdependent noise level
Note that this is equivalent to adding a WhiteKernel with calpha Allowing to specify the
noise level directly as a parameter is mainly for convenience and for consistency with Ridge
optimizer string or callable optional default “fminlbfgsb” Can either be one of the
internally supported optimizers for optimizing the kernel’s parameters specified by a string
or an externally defined optimizer passed as a callable If a callable is passed it must have
the signature

```
defoptimizerobjfunc initialtheta bounds
objfunc is the objective function to be minimized which
takes the hyperparameters theta as parameter and an
optional flag evalgradient which determines if the
gradient is returned additionally to the function value
initialtheta the initial value for theta which can be
used by local optimizers
bounds the bounds on the values of theta
```

Returned are the best found hyperparameters theta and
the corresponding value of the target function

returnthetaopt funcmin

Per default the ‘fminlbfgsb’ algorithm from scipyoptimize is used If None is passed
the kernel’s parameters are kept fixed Available internal optimizers are

617sklearnprocess Gaussian Processes 1803

fminlbfgsb

nrestartsoptimizer int optional default 0 The number of restarts of the optimizer for finding the kernel’s parameters which maximize the logmarginal likelihood The first run of the optimizer is performed from the kernel’s initial parameters the remaining ones if any from thetas sampled loguniform randomly from the space of allowed thetavalues If greater than 0 all bounds must be finite Note that nrestartsoptimizer 0 implies that one run is performed

normalizey boolean optional default False Whether the target values y are normalized ie the mean of the observed target values become zero This parameter should be set to True if the target values’ mean is expected to differ considerable from zero When enabled the normalization effectively modifies the GP’s prior based on the data which contradicts the likelihood principle normalization is thus disabled per default

copyXtrain bool optional default True If True a persistent copy of the training data is stored in the object Otherwise just a reference to the training data is stored which might cause predictions to change if the data is modified externally

randomstate int RandomState instance or None optional default None The generator used to initialize the centers If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Attributes

Xtrain arraylike shape nsamples nfeatures Feature values in training data also required for prediction

ytrain arraylike shape nsamples noutputdims Target values in training data also required for prediction

kernel kernel object The kernel used for prediction The structure of the kernel is the same as the one passed as parameter but with optimized hyperparameters

Larraylike shape nsamples nsamples Lowertriangular Cholesky decomposition of the kernel in Xtrain

alpha arraylike shape nsamples Dual coefficients of training data points in kernel space

logmarginallikelihoodvalue float The logmarginallikelihood of selfkernel

theta

Examples

```
from sklearndatasets import makefriedman2
from sklearngaussianprocess import GaussianProcessRegressor
from sklearngaussianprocesskernels import DotProduct WhiteKernel
X y makefriedman2nsamples500 noise0 randomstate0
kernel DotProduct WhiteKernel
gpr GaussianProcessRegressorkernelkernel
randomstate0fitX y
gprscoreX y
03680
gprpredictX2 returnstd True
array6530 5921 array3166 3166
```

scikitlearn user guide Release 0213

Methods

fitself X y Fit Gaussian process regression model
getparams self deep Get parameters for this estimator
logmarginallikelihood self theta Returns logmarginal likelihood of theta for training data
predict self X returnstd returncov Predict using the Gaussian process regression model
sample self X nsamples randomstate Draw samples from Gaussian process and evaluate at X
score self X y sampleweight Returns the coefficient of determination R2 of the prediction
setparams self params Set the parameters of this estimator
init selfkernelNone alpha1e10 optimizer'fminlbfgsb' nrestartsoptimizer0 normalizeFalse copyXtrainTrue randomstateNone
fitselfXy
Fit Gaussian process regression model
Parameters
Xarraylike shape nsamples nfeatures Training data
yarraylike shape nsamples noutputdims Target values
Returns
self returns an instance of self
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values
logmarginallikelihood selfthetaNone evalgradientFalse
Returns logmarginal likelihood of theta for training data
Parameters
theta arraylike shape nkernelparams or None Kernel hyperparameters for which the logmarginal likelihood is evaluated If None the precomputed logmarginallikelihood of selfkerneltheta is returned
evalgradient bool default False If True the gradient of the logmarginal likelihood with respect to the kernel hyperparameters at position theta is returned additionally If True theta must not be None
Returns
loglikelihood float Logmarginal likelihood of theta for training data
loglikelihoodgradient array shape nkernelparams optional Gradient of the log marginal likelihood with respect to the kernel hyperparameters at position theta Only returned when evalgradient is True
617sklearn Gaussian Processes 1805

scikitlearn user guide Release 0213

predictselfXreturnstdFalse returncovFalse

Predict using the Gaussian process regression model

We can also predict based on an unfitted model by using the GP prior In addition to the mean of the predictive distribution also its standard deviation returnstdTrue or covariance returncovTrue Note that at most one of the two can be requested

Parameters

Xarraylike shape nsamples nfeatures Query points where the GP is evaluated

returnstd bool default False If True the standard deviation of the predictive distribution at the query points is returned along with the mean

returncov bool default False If True the covariance of the joint predictive distribution at the query points is returned along with the mean

Returns

ymean array shape nsamples noutputdims Mean of predictive distribution at query points

ystd array shape nsamples optional Standard deviation of predictive distribution at query points Only returned when returnstd is True

ycov array shape nsamples nsamples optional Covariance of joint predictive distribution at query points Only returned when returncov is True

sample selfXnsamples1 randomstate0

Draw samples from Gaussian process and evaluate at X

Parameters

Xarraylike shape nsamplesX nfeatures Query points where the GP samples are evaluated

nsamples int default 1 The number of samples drawn from the Gaussian process

randomstate int RandomState instance or None optional default0 If int randomstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Returns

ysamples array shape nsamplesX noutputdims nsamples Values of

nsamples samples drawn from Gaussian process and evaluated at query points

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

1806 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearngaussianprocessGaussianProcessRegressor

- Comparison of kernel ridge and Gaussian process regression

- Illustration of prior and posterior Gaussian process for different kernels

- Gaussian process regression GPR with noiselevel estimation

- Gaussian Processes regression basic introductory example

- Gaussian process regression GPR on Mauna Loa CO2 data

Kernels

gaussianprocesskernels

CompoundKernel kernelsKernel which is composed of a set of other kernels

gaussianprocesskernels

ConstantKernel Constant kernel

gaussianprocesskernels

DotProduct DotProduct kernel

gaussianprocesskernels

ExpSineSquared ExpSineSquared kernel

gaussianprocesskernels

Exponentiation Exponentiate kernel by given exponent

gaussianprocesskernelsHyperparameter A kernel hyperparameter's specification in form of a

namedtuple

gaussianprocesskernelsKernel Base class for all kernels

gaussianprocesskernelsMatern Matern kernel

gaussianprocesskernels

PairwiseKernel Wrapper for kernels in sklearnmetricspairwise

gaussianprocesskernelsProduct k1 k2 Productkernel k1 k2 of two kernels k1 and k2

Continued on next page

617sklearngaussianprocess Gaussian Processes 1807

scikitlearn user guide Release 0213

Table 6119 – continued from previous page

gaussianprocesskernelsRBF lengthscale

Radialbasis function kernel aka squaredexponential ker
nel

gaussianprocesskernels

RationalQuadratic Rational Quadratic kernel

gaussianprocesskernelsSum k1 k2 Sumkernel k1 k2 of two kernels k1 and k2

gaussianprocesskernels

WhiteKernel White kernel

6173sklearn gaussianprocesskernels CompoundKernel

class sklearn gaussianprocesskernels CompoundKernel kernels

Kernel which is composed of a set of other kernels

New in version 018

Parameters

kernels list of Kernel objects The other kernels

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameters Returns a list of all hyperparameter specifications

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selfkernels

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

Note that this compound kernel returns the results of all simple kernel stacked along an additional axis

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the
returned kernel kX Y If None kX X if evaluated instead

evalgradient bool optional defaultFalse Determines whether the gradient with re
spect to the kernel hyperparameter is determined

Returns

1808 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Karray shape nsamplesX nsamplesY nkernels Kernel kX Y

Kgradient array shape nsamplesX nsamplesX ndims nkernels The gradient of the kernel kX X with respect to the hyperparameter of the kernel Only returned when evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparam

eters theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Returns

Kdiag array shape nsamplesX nkernels Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

617sklearn gaussianprocess Gaussian Processes 1809

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel’s hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

6174sklearn Gaussian process kernels ConstantKernel

class sklearn Gaussian process kernels ConstantKernel constantvalue10

constantvaluebounds1e

051000000

Constant kernel

Can be used as part of a productkernel where it scales the magnitude of the other factor kernel or as part of a sumkernel where it modifies the mean of the Gaussian process

kx1 x2 constantvalue for all x1 x2

New in version 018

Parameters

constantvalue float default 10 The constant value which defines the covariance kx1

x2 constantvalue

constantvaluebounds pair of floats 0 default 1e5 1e5 The lower and upper bound on constantvalue

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameterconstantvalue

hyperparameters Returns a list of all hyperparameter specifications

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selfconstantvalue10 constantvaluebounds1e05 1000000

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

scikitlearn user guide Release 0213

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the returned kernel kX Y If None kX X if evaluated instead

evalgradient bool optional defaultFalse Determines whether the gradient with respect to the kernel hyperparameter is determined Only supported when Y is None

Returns

Karray shape nsamplesX nsamplesY Kernel kX Y

Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the kernel kX X with respect to the hyperparameter of the kernel Only returned when evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparameters

theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

617sklearn gaussianprocess Gaussian Processes 1811

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

Examples using sklearnprocesskernelsConstantKernel

- Illustration of prior and posterior Gaussian process for different kernels
- Isoprobability lines for Gaussian Processes classification GPC
- Gaussian Processes regression basic introductory example

6175sklearnprocesskernels DotProduct

classsklearnprocesskernels DotProduct sigma010 sigma0bounds1e051000000

DotProduct kernel

The DotProduct kernel is nonstationary and can be obtained from linear regression by putting N0 1 priors on the coefficients of $x^T d$ and a prior of N0 sigma02 on the bias The DotProduct

kernel is invariant to a rotation of the coordinates about the origin but not translations It is parameterized by a parameter sigma02 For sigma02 0 the kernel is called the homogeneous linear kernel otherwise it is

inhomogeneous The kernel is given by

$$k(x_i, x_j) = \sigma_0 + \frac{\sigma_0^2}{2} x_i^T x_j$$

The DotProduct kernel is commonly combined with exponentiation

New in version 018

Parameters

sigma0 float 0 default 10 Parameter controlling the inhomogeneity of the kernel If

sigma00 the kernel is homogenous

sigma0bounds pair of floats 0 default 1e5 1e5 The lower and upper bound on l

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameterssigma0

hyperparameters Returns a list of all hyperparameter specifications

ndims Returns the number of nonfixed hyperparameters of the kernel

1812 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selfsigma010 sigma0bounds1e05 1000000

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the

returned kernel kX Y If None kX X if evaluated instead

evalgradient bool optional defaultFalse Determines whether the gradient with re
spect to the kernel hyperparameter is determined Only supported when Y is None

Returns

Karray shape nsamplesX nsamplesY Kernel kX Y

Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the
kernel kX X with respect to the hyperparameter of the kernel Only returned when
evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel’s hyperparam
eters theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently
since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Returns

617sklearn gaussianprocess Gaussian Processes 1813

scikitlearn user guide Release 0213

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

Examples using sklearngaussianprocesskernelsDotProduct

- Illustration of Gaussian process classification GPC on the XOR dataset
- Illustration of prior and posterior Gaussian process for different kernels
- Isoprobability lines for Gaussian Processes classification GPC

6176sklearngaussianprocesskernels ExpSineSquared

classsklearngaussianprocesskernels ExpSineSquared lengthscale10

periodicity10

lengthscalebounds1e

05 1000000

periodicitybounds1e05

1000000

ExpSineSquared kernel

1814 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The ExpSineSquared kernel allows modeling periodic functions It is parameterized by a lengthscale parameter lengthscale0 and a periodicity parameter periodicity0 Only the isotropic variant where l is a scalar is supported at the moment The kernel given by

$$k(x_i, x_j) = \exp\left(-\frac{2}{\text{lengthscale}} \sin\left(\frac{\text{periodicity}}{2} |x_i - x_j|\right)\right)$$

New in version 0.18

Parameters

- lengthscale float, 0 default 10 The length scale of the kernel
- periodicity float, 0 default 10 The periodicity of the kernel
- lengthscalebounds pair of floats, 0 default 1e5 1e5 The lower and upper bound on lengthscale
- periodicitybounds pair of floats, 0 default 1e5 1e5 The lower and upper bound on periodicity

Attributes

- bounds Returns the logtransformed bounds on the theta
- hyperparameterlengthscale
- hyperparameterperiodicity
- hyperparameters Returns a list of all hyperparameter specifications
- ndims Returns the number of nonfixed hyperparameters of the kernel
- theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

- call self X Y evalgradient Return the kernel k(X, Y) and optionally its gradient
 - clonewiththeta self theta Returns a clone of self with given hyperparameters theta
 - diag self X Returns the diagonal of the kernel k(X, X)
 - getparams self deep Get parameters of this kernel
 - isstationary self Returns whether the kernel is stationary
 - setparams self params Set the parameters of this kernel
 - init selflengthscale10 periodicity10 lengthscalebounds1e05 1000000 periodicitybounds1e05 1000000
 - call selfXYNone evalgradientFalse
- Return the kernel k(X, Y) and optionally its gradient

Parameters

- Xarray shape nsamplesX nfeatures Left argument of the returned kernel k(X, Y)
 - Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the returned kernel k(X, Y) If None k(X, X) if evaluated instead
 - evalgradient bool optional defaultFalse Determines whether the gradient with respect to the kernel hyperparameter is determined Only supported when Y is None
- Returns
- 617sklearn.gaussianprocess Gaussian Processes 1815

scikitlearn user guide Release 0213

Karray shape nsamplesX nsamplesY Kernel kX Y

Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the kernel kX X with respect to the hyperparameter of the kernel Only returned when evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparam

eters theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

scikitlearn user guide Release 0213

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel’s hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

Examples using sklearngaussianprocesskernelsExpSineSquared

- Comparison of kernel ridge and Gaussian process regression
- Illustration of prior and posterior Gaussian process for different kernels
- Gaussian process regression GPR on Mauna Loa CO2 data

6177sklearngaussianprocesskernels Exponentiation

classsklearngaussianprocesskernels Exponentiation kernel exponent

Exponentiate kernel by given exponent

The resulting kernel is defined as $k_{exp}(X, Y) = k(X, Y)^{exponent}$

New in version 018

Parameters

kernel Kernel object The base kernel

exponent float The exponent for the base kernel

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameters Returns a list of all hyperparameter

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel $k(X, Y)$ and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel $k(X, X)$

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selfkernel exponent

call selfXYNone evalgradientFalse

Return the kernel $k(X, Y)$ and optionally its gradient

617sklearngaussianprocess Gaussian Processes 1817

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y
Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the returned kernel kX Y If None kX X if evaluated instead
evalgradient bool optional defaultFalse Determines whether the gradient with respect to the kernel hyperparameter is determined

Returns

Karray shape nsamplesX nsamplesY Kernel kX Y
Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the kernel kX X with respect to the hyperparameter of the kernel Only returned when evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparameters theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X
The result of this method is identical to npdiagselfX however it can be evaluated more efficiently since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

6178sklearn Gaussian Process kernels Hyperparameter

class sklearn Gaussian Process kernels Hyperparameter

A kernel hyperparameter's specification in form of a namedtuple

New in version 0.18

Attributes

name string Alias for field number 0

valuetype string Alias for field number 1

bounds pair of floats 0 or "fixed" Alias for field number 2

nelements int default 1 Alias for field number 3

fixed bool default None Alias for field number 4

Methods

count

index Raises ValueError if the value is not present

init selfargs kwargs

Initialize self See helptypeself for accurate signature

call args kwargs

Call self as a function

bounds

Alias for field number 2

count

fixed

Alias for field number 4

index

6178sklearn Gaussian Process Gaussian Processes 1819

scikitlearn user guide Release 0213
Raises ValueError if the value is not present
nelements
Alias for field number 3
name
Alias for field number 0
valuetype
Alias for field number 1
6179sklearngaussianprocesskernels Kernel
classssklearngaussianprocesskernels Kernel
Base class for all kernels
New in version 018
Attributes
bounds Returns the logtransformed bounds on the theta
hyperparameters Returns a list of all hyperparameter specifications
ndims Returns the number of nonfixed hyperparameters of the kernel
theta Returns the flattened logtransformed nonfixed hyperparameters
Methods
call self X Y evalgradient Evaluate the kernel
clonewiththeta self theta Returns a clone of self with given hyperparameters
theta
diag self X Returns the diagonal of the kernel kX X
getparams self deep Get parameters of this kernel
isstationary self Returns whether the kernel is stationary
setparams self params Set the parameters of this kernel
init selfargs kwargs
Initialize self See helptypeself for accurate signature
call selfXYNone evalgradientFalse
Evaluate the kernel
bounds
Returns the logtransformed bounds on the theta
Returns
bounds array shape ndims 2 The logtransformed bounds on the kernel’s hyperparam
eters theta
clonewiththeta selftheta
Returns a clone of self with given hyperparameters theta
Parameters
theta array shape ndims The hyperparameters
1820 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

diagselfX

Returns the diagonal of the kernel $k \times X$

The result of this method is identical to `npdiagselfX` however it can be evaluated more efficiently since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel $k \times Y$

Returns

Kdiag array shape nsamplesX Diagonal of kernel $k \times X$

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

61710sklearn gaussianprocess kernels Matern

class sklearn gaussianprocess kernels Matern lengthscale10 lengthscale bounds1e

051000000 nu15

Matern kernel

The class of Matern kernels is a generalization of the RBF (Radial Basis Function) and the absolute exponential kernel parameterized by an additional parameter nu The smaller nu the less smooth the approximated function is For nuinf the kernel

617sklearn gaussianprocess Gaussian Processes 1821

scikitlearn user guide Release 0213

becomes equivalent to the RBF (Radial Basis Function) kernel and for nu05 to the absolute exponential kernel Important intermediate values are nu15 once differentiable functions and nu25 twice differentiable functions

See Rasmussen and Williams 2006 pp84 for details regarding the different variants of the Matern kernel

New in version 018

Parameters

lengthscale float or array with shape nfeatures default 10 The length scale of the kernel

If a float an isotropic kernel is used If an array an anisotropic kernel is used where each dimension of l defines the lengthscale of the respective feature dimension

lengthscalebounds pair of floats 0 default 1e5 1e5 The lower and upper bound on lengthscale

nu float default 15 The parameter nu controlling the smoothness of the learned function

The smaller nu the less smooth the approximated function is For nuinf the kernel becomes equivalent to the RBF (Radial Basis Function) kernel and for nu05 to the absolute exponential kernel Important intermediate values are nu15 once differentiable functions and nu25 twice differentiable functions Note that values of nu not in 05 15 25 inf incur a considerably higher computational cost approx 10 times higher since they require to evaluate the modified Bessel function Furthermore in contrast to l nu is kept fixed to its initial value and not optimized

Attributes

anisotropic

bounds Returns the logtransformed bounds on the theta

hyperparameterlengthscale

hyperparameters Returns a list of all hyperparameter specifications

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selflengthscale10 lengthscalebounds1e05 1000000 nu15

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the returned kernel kX Y If None kX X if evaluated instead

1822 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

evalgradient bool optional defaultFalse Determines whether the gradient with respect to the kernel hyperparameter is determined Only supported when Y is None
Returns

Karray shape nsamplesX nsamplesY Kernel kX Y
Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the kernel kX X with respect to the hyperparameter of the kernel Only returned when evalgradient is True
bounds

Returns the logtransformed bounds on the theta

Returns
bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparameters theta

clonewiththeta selftheta
Returns a clone of self with given hyperparameters theta

Parameters
theta array shape ndims The hyperparameters
diagselfX

Returns the diagonal of the kernel kX X
The result of this method is identical to npdiagselfX however it can be evaluated more efficiently since only the diagonal is evaluated
Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y
Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X
getparams selfdeepTrue

Get parameters of this kernel
Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators
Returns

params mapping of string to any Parameter names mapped to their values
hyperparameters

Returns a list of all hyperparameter specifications
isstationary self

Returns whether the kernel is stationary
ndims

Returns the number of nonfixed hyperparameters of the kernel
setparams selfparams

Set the parameters of this kernel
The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object
617sklearn gaussianprocess Gaussian Processes 1823

scikitlearn user guide Release 0213

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel’s hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

Examples using sklearn.gaussianprocess.kernels.Matern

• Illustration of prior and posterior Gaussian process for different kernels

61711sklearn.gaussianprocess.kernels.PairwiseKernel

class sklearn.gaussianprocess.kernels.PairwiseKernel gamma=10

gamma_bounds=(1e-05,

1000000) metric='linear' pair

wis_kernel_kwargs=None

Wrapper for kernels in sklearn.metrics.pairwise

A thin wrapper around the functionality of the kernels in sklearn.metrics.pairwise

Note Evaluation of eval_gradient is not analytic but numeric and all kernels support only isotropic distances The parameter gamma is considered to be a hyperparameter and may be optimized The other kernel parameters are set directly at initialization and are kept fixed

New in version 0.18

Parameters

gamma float [0, default 10] Parameter gamma of the pairwise kernel specified by metric

gamma_bounds pair of floats [0, default (1e-05, 1e5)] The lower and upper bound on

gamma
metric string or callable [default “linear”] The metric to use when calculating kernel between instances in a feature array If metric is a string it must be one of the metrics in pairwise.PAIRWISE_KERNEL_FUNCTIONS If metric is “precomputed” X is assumed to be a kernel matrix Alternatively if metric is a callable function it is called on each pair of instances rows and the resulting value recorded The callable should take two arrays from X as input and return a value indicating the distance between them

pairwise_kernel_kwargs dict [default None] All entries of this dict if any are passed as keyword arguments to the pairwise kernel function

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameter_gamma

hyperparameters Returns a list of all hyperparameter specifications

ndims Returns the number of nonfixed hyperparameters of the kernel

1824 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selfgamma10 gammabounds1e05 1000000 metric'linear' pair

wisekernelskwargsNone

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the

returned kernel kX Y If None kX X if evaluated instead

evalgradient bool optional defaultFalse Determines whether the gradient with re
spect to the kernel hyperparameter is determined Only supported when Y is None

Returns

Karray shape nsamplesX nsamplesY Kernel kX Y

Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the
kernel kX X with respect to the hyperparameter of the kernel Only returned when

evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparam

eters theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently
since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

617sklearnGaussianProcess Gaussian Processes 1825

scikitlearn user guide Release 0213

Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form

componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representa

tion of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales

naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

61712sklearn gaussianprocess kernels Product

class sklearn gaussianprocess kernels Productk1k2

Productkernel k1 k2 of two kernels k1 and k2

The resulting kernel is defined as kprodX Y k1X Y k2X Y

New in version 018

Parameters

k1Kernel object The first basekernel of the productkernel

k2Kernel object The second basekernel of the productkernel

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameters Returns a list of all hyperparameter

1826 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selfk1k2

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the

returned kernel kX Y If None kX X if evaluated instead

evalgradient bool optional defaultFalse Determines whether the gradient with re

spect to the kernel hyperparameter is determined

Returns

Karray shape nsamplesX nsamplesY Kernel kX Y

Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the

kernel kX X with respect to the hyperparameter of the kernel Only returned when

evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparam

eters theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently

since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

617sklearn gaussianprocess Gaussian Processes 1827

scikitlearn user guide Release 0213

Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form

componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representa

tion of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales

naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

61713sklearn Gaussian process kernels RBF (Radial Basis Function)

class sklearn Gaussian process kernels RBF lengthscale10 lengthscale bounds1e05

1000000

Radial basis function kernel aka squared exponential kernel

The RBF (Radial Basis Function) kernel is a stationary kernel It is also known as the "squared exponential" kernel It is parameterized

by a lengthscale parameter lengthscale0 which can either be a scalar isotropic variant of the kernel or a

vector with the same number of dimensions as the inputs X anisotropic variant of the kernel The kernel is

given by

$k(x_i, x_j) = \exp(-\frac{1}{2} \sum_{d=1}^D \frac{(x_{i,d} - x_{j,d})^2}{\text{lengthscale}_d^2})$

This kernel is infinitely differentiable which implies that GPs with this kernel as covariance function have mean

square derivatives of all orders and are thus very smooth

New in version 0.18

1828 Chapter 6 API (Application Programming Interface) Reference

Parameters

lengthscale float or array with shape nfeatures default 10 The length scale of the kernel
If a float an isotropic kernel is used If an array an anisotropic kernel is used where each
dimension of l defines the lengthscale of the respective feature dimension
lengthscalebounds pair of floats 0 default 1e5 1e5 The lower and upper bound on

lengthscale

Attributes

anisotropic

bounds Returns the logtransformed bounds on the theta

hyperparameterlengthscale

hyperparameters Returns a list of all hyperparameter specifications

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selflengthscale10 lengthscalebounds1e05 1000000

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the

returned kernel kX Y If None kX X if evaluated instead

evalgradient bool optional defaultFalse Determines whether the gradient with re
spect to the kernel hyperparameter is determined Only supported when Y is None

Returns

Karray shape nsamplesX nsamplesY Kernel kX Y

Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the
kernel kX X with respect to the hyperparameter of the kernel Only returned when
evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

617sklearn gaussianprocess Gaussian Processes 1829

scikitlearn user guide Release 0213

bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparameters theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

1830 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples using sklearn.gaussianprocess.kernels.RBF

- Plot classification probability
- Classifier comparison
- Illustration of Gaussian process classification GPC on the XOR dataset
- Gaussian process classification GPC on iris dataset
- Illustration of prior and posterior Gaussian process for different kernels
- Probabilistic predictions with Gaussian process classification GPC
- Gaussian process regression GPR with noiselevel estimation
- Gaussian Processes regression basic introductory example
- Gaussian process regression GPR on Mauna Loa CO2 data

61714sklearn.gaussianprocess.kernels.RationalQuadratic
classsklearn.gaussianprocess.kernels.RationalQuadratic lengthscale=10
alpha=10

lengthscale.bounds=(1e

05, 1000000)

alpha.bounds=(1e05,

1000000)

Rational Quadratic kernel

The RationalQuadratic kernel can be seen as a scale mixture an infinite sum of RBF (Radial Basis Function) kernels with different characteristic lengthscales. It is parameterized by a lengthscale parameter lengthscale=0 and a scale mixture parameter alpha=0. Only the isotropic variant where lengthscale is a scalar is supported at the moment. The kernel given by

$$k(x_i, x_j) = \frac{1}{\alpha} \sum_{j=1}^{\infty} \frac{\alpha^j}{2^j j!} \exp\left(-\frac{\alpha^j \|x_i - x_j\|^2}{2^j}\right)$$

New in version 0.18

Parameters

lengthscale float, 0 default 10 The length scale of the kernel

alpha float, 0 default 10 Scale mixture parameter

lengthscale.bounds pair of floats, 0 default (1e5, 1e5) The lower and upper bound on lengthscale

alpha.bounds pair of floats, 0 default (1e5, 1e5) The lower and upper bound on alpha

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameter.alpha

hyperparameter.lengthscale

hyperparameters Returns a list of all hyperparameter specifications

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

617sklearn.gaussianprocess Gaussian Processes 1831

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient
clonewiththeta self theta Returns a clone of self with given hyperparameters

theta
diag self X Returns the diagonal of the kernel kX X
getparams self deep Get parameters of this kernel
isstationary self Returns whether the kernel is stationary
setparams self params Set the parameters of this kernel
init self lengthscale10 alpha10 lengthscalebounds1e05 1000000
alphabounds1e05 1000000
call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient
Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y
Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the
returned kernel kX Y If None kX X if evaluated instead
evalgradient bool optional defaultFalse Determines whether the gradient with re
spect to the kernel hyperparameter is determined Only supported when Y is None
Returns

Karray shape nsamplesX nsamplesY Kernel kX Y
Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the
kernel kX X with respect to the hyperparameter of the kernel Only returned when
evalgradient is True

bounds
Returns the logtransformed bounds on the theta
Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel’s hyperparam
eters theta
clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta
Parameters

theta array shape ndims The hyperparameters
diagselfX

Returns the diagonal of the kernel kX X
The result of this method is identical to npdiagselfX however it can be evaluated more efficiently
since only the diagonal is evaluated
Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y
Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X
1832 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

Examples using sklearngaussianprocesskernelsRationalQuadratic

•Illustration of prior and posterior Gaussian process for different kernels

•Gaussian process regression GPR on Mauna Loa CO2 data

61715sklearngaussianprocesskernels Sum

classsklearngaussianprocesskernels Sumk1k2

Sumkernel k1 k2 of two kernels k1 and k2

The resulting kernel is defined as $k_{sum}(X, Y) = k_1(X, Y) + k_2(X, Y)$

New in version 018

Parameters

k1Kernel object The first basekernel of the sumkernel

k2Kernel object The second basekernel of the sumkernel

617sklearngaussianprocess Gaussian Processes 1833

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameters Returns a list of all hyperparameter

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters

theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selfk1k2

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the returned kernel kX Y If None kX X if evaluated instead

evalgradient bool optional defaultFalse Determines whether the gradient with respect to the kernel hyperparameter is determined

Returns

Karray shape nsamplesX nsamplesY Kernel kX Y

Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the kernel kX X with respect to the hyperparameter of the kernel Only returned when evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel's hyperparameters theta

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

scikitlearn user guide Release 0213

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

61716sklearn Gaussian process kernels WhiteKernel

class sklearn Gaussian process kernels WhiteKernel noiselevel10

noiselevelbounds1e05

1000000

White kernel

The main usecase of this kernel is as part of a sumkernel where it explains the noise component of the signal Tuning its parameter corresponds to estimating the noiselevel

kx1 x2 noiselevel if x1 x2 else 0

617sklearn Gaussian process Gaussian Processes 1835

scikitlearn user guide Release 0213

New in version 018

Parameters

noiselevel float default 10 Parameter controlling the noise level

noiselevelbounds pair of floats 0 default 1e5 1e5 The lower and upper bound on noiselevel

Attributes

bounds Returns the logtransformed bounds on the theta

hyperparameternoiselevel

hyperparameters Returns a list of all hyperparameter specifications

ndims Returns the number of nonfixed hyperparameters of the kernel

theta Returns the flattened logtransformed nonfixed hyperparameters

Methods

call self X Y evalgradient Return the kernel kX Y and optionally its gradient

clonewiththeta self theta Returns a clone of self with given hyperparameters theta

diag self X Returns the diagonal of the kernel kX X

getparams self deep Get parameters of this kernel

isstationary self Returns whether the kernel is stationary

setparams self params Set the parameters of this kernel

init selfnoiselevel10 noiselevelbounds1e05 1000000

call selfXYNone evalgradientFalse

Return the kernel kX Y and optionally its gradient

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Yarray shape nsamplesY nfeatures optional defaultNone Right argument of the

returned kernel kX Y If None kX X if evaluated instead

evalgradient bool optional defaultFalse Determines whether the gradient with respect to the kernel hyperparameter is determined Only supported when Y is None

Returns

Karray shape nsamplesX nsamplesY Kernel kX Y

Kgradient array opt shape nsamplesX nsamplesX ndims The gradient of the kernel kX X with respect to the hyperparameter of the kernel Only returned when evalgradient is True

bounds

Returns the logtransformed bounds on the theta

Returns

bounds array shape ndims 2 The logtransformed bounds on the kernel’s hyperparameters theta

1836 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

clonewiththeta selftheta

Returns a clone of self with given hyperparameters theta

Parameters

theta array shape ndims The hyperparameters

diagselfX

Returns the diagonal of the kernel kX X

The result of this method is identical to npdiagselfX however it can be evaluated more efficiently since only the diagonal is evaluated

Parameters

Xarray shape nsamplesX nfeatures Left argument of the returned kernel kX Y

Returns

Kdiag array shape nsamplesX Diagonal of kernel kX X

getparams selfdeepTrue

Get parameters of this kernel

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

hyperparameters

Returns a list of all hyperparameter specifications

isstationary self

Returns whether the kernel is stationary

ndims

Returns the number of nonfixed hyperparameters of the kernel

setparams selfparams

Set the parameters of this kernel

The method works on simple kernels as well as on nested kernels The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

theta

Returns the flattened logtransformed nonfixed hyperparameters

Note that theta are typically the logtransformed values of the kernel's hyperparameters as this representation of the search space is more amenable for hyperparameter search as hyperparameters like lengthscales naturally live on a logscale

Returns

theta array shape ndims The nonfixed logtransformed hyperparameters of the kernel

617sklearn gaussianprocess Gaussian Processes 1837

scikitlearn user guide Release 0213

Examples using sklearn.gaussianprocess.kernels.WhiteKernel

- Comparison of kernel ridge and Gaussian process regression
- Gaussian process regression GPR with noiselevel estimation
- Gaussian process regression GPR on Mauna Loa CO2 data

618sklearn.isotonic Isotonic regression

User guide See the Isotonic regression section for further details

isotonicIsotonicRegression ymin ymax

Isotonic regression model

6181sklearn.isotonic IsotonicRegression

classsklearn.isotonic IsotonicRegression ymin=None ymax=None increasing=True

outofbounds='nan'

Isotonic regression model

The isotonic regression optimization problem is defined by

$\min \sum w_i (y_i - \hat{y}_i)^2$

subject to $\hat{y}_i \leq \hat{y}_j$ whenever $X_i \leq X_j$

and $\min y \leq \hat{y} \leq \max y$

where

- y_i are inputs real numbers
- \hat{y}_i are fitted
- X specifies the order If X is nondecreasing then y is nondecreasing
- w_i are optional strictly positive weights default to 10

Read more in the User Guide

Parameters

ymin optional default None If not None set the lowest value of the fit to ymin

ymax optional default None If not None set the highest value of the fit to ymax

increasing boolean or string optional default True If boolean whether or not to fit the

isotonic regression with y increasing or decreasing

The string value "auto" determines whether y should increase or decrease based on the

Spearman correlation estimate's sign

outofbounds string optional default "nan" The outofbounds parameter handles

how xvalues outside of the training domain are handled When set to "nan" predicted

yvalues will be NaN When set to "clip" predicted yvalues will be set to the value corre

sponding to the nearest train interval endpoint When set to "raise" allow interp1d to

throw ValueError

1838 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Attributes

Xmin float Minimum value of input array Xfor left bound

Xmax float Maximum value of input array Xfor right bound

ffunction The stepwise interpolating function that covers the input domain X

Notes

Ties are broken using the secondary method from Leeuw 1977

References

Isotonic Median Regression A Linear Programming Approach Nilotpal Chakravarti Mathematics of Operations

Research V ol 14 No 2 May 1989 pp 303308

Isotone Optimization in R PoolAdjacentViolators Algorithm PA V A and Active Set Methods Leeuw Hornik

Mair Journal of Statistical Software 2009

Correctness of Kruskal’s algorithms for monotone regression with ties Leeuw Psychometrika 1977

Examples

from sklearndatasets import makeregression

from sklearnisotonic import IsotonicRegression

X y makeregressionnsamples10 nfeatures1 randomstate41

isoreg IsotonicRegressionfitXflatten y

isoregpredict1 2

array18628 37256

Methods

fitself X y sampleweight Fit the model using X y as training data

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

predict self T Predict new data by linear interpolation

score self X y sampleweight Returns the coefficient of determination R2 of the pre
diction

setparams self params Set the parameters of this estimator

transform self T Transform new data by linear interpolation

init selfyminNone ymaxNone increasingTrue outofbounds’nan’

fitselfXysampleweightNone

Fit the model using X y as training data

Parameters

Xarraylike shapensamples Training data

yarraylike shapensamples Training target

618sklearnisotonic Isotonic regression 1839

scikitlearn user guide Release 0213

sampleweight arraylike shapensamples optional default None Weights If set to None all weights will be set to 1 equal weights

Returns

self object Returns an instance of self

Notes

X is stored for future use as transform needs X to interpolate new input data

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfT

Predict new data by linear interpolation

Parameters

Tarraylike shapensamples Data to transform

Returns

Tarray shapensamples Transformed data

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

1840 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfT

Transform new data by linear interpolation

Parameters

Tarraylike shapensamples Data to transform

Returns

Tarray shapensamples The transformed data

Examples using sklearnisotonicIsotonicRegression

•Isotonic Regression

isotoniccheckincreasing x y Determine whether y is monotonically correlated with x

isotonicisotonicregression y Solve the isotonic regression model

6182sklearnisotonic checkincreasing

sklearnisotonic checkincreasing xy

Determine whether y is monotonically correlated with x

y is found increasing or decreasing with respect to x based on a Spearman correlation test

Parameters

xarraylike shapensamples Training data

yarraylike shapensamples Training target

Returns

618sklearnisotonic Isotonic regression 1841

scikitlearn user guide Release 0213

increasingbool boolean Whether the relationship is increasing or decreasing

Notes

The Spearman correlation coefficient is estimated from the data and the sign of the resulting estimate is used as the result

In the event that the 95 confidence interval based on Fisher transform spans zero a warning is raised

References

Fisher transformation Wikipedia https://en.wikipedia.org/wiki/Fisher_transform

6183sklearnisotonic isotonicregression

sklearnisotonic isotonicregression ysampleweightNone yminNone ymaxNone

increasingTrue

Solve the isotonic regression model

min sum $w_i (y_i - \hat{y}_i)^2$

subject to $y_{\min} \leq y_1 \leq y_2 \leq \dots \leq y_n \leq y_{\max}$

where

- y_i are inputs real numbers
- \hat{y}_i are fitted
- w_i are optional strictly positive weights default to 1

Read more in the User Guide

Parameters

yiterable of floats The data

sampleweight iterable of floats optional default None Weights on each point of the regres

sion If None weight is set to 1 equal weights

ymin optional default None If not None set the lowest value of the fit to ymin

ymax optional default None If not None set the highest value of the fit to ymax

increasing boolean optional default True Whether to compute y_i s increasing if set to

True or decreasing if set to False

Returns

ylist of floats Isotonic fit of y

References

“Active set algorithms for isotonic regression A unifying framework” by Michael J Best and Nilotpal

Chakravarti section 3

1842 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

619sklearnimpute Impute

Transformers for missing value imputation

User guide See the Imputation of missing values section for further details

imputeSimpleImputer missingvalues Imputation transformer for completing missing values

imputeliterativeImputer estimator Multivariate imputer that estimates each feature from all the others

imputeMissingIndicator missingvalues Binary indicators for missing values

6191sklearnimpute SimpleImputer

classssklearnimpute SimpleImputer missingvaluesnan strategy‘mean’ fillvalueNone ver

bose0 copyTrue addindicatorFalse

Imputation transformer for completing missing values

Read more in the User Guide

Parameters

missingvalues number string nnan default or None The placeholder for the missing

values All occurrences of missingvalues will be imputed

strategy string optional default“mean” The imputation strategy

- If “mean” then replace missing values using the mean along each column Can only be used with numeric data

- If “median” then replace missing values using the median along each column Can only be used with numeric data

- If “mostfrequent” then replace missing using the most frequent value along each column Can be used with strings or numeric data

- If “constant” then replace missing values with fillvalue Can be used with strings or numeric data

New in version 020 strategy“constant” for fixed value imputation

fillvalue string or numerical value optional defaultNone When strategy “constant”

fillvalue is used to replace all occurrences of missingvalues If left to the default fillvalue will be 0 when imputing numerical data and “missingvalue” for strings or object data types

verbose integer optional default0 Controls the verbosity of the imputer

copy boolean optional defaultTrue If True a copy of X will be created If False imputation will be done inplace whenever possible Note that in the following cases a new copy will

always be made even if copyFalse

- If X is not an array of floating values

- If X is encoded as a CSR (Compressed Sparse Row) matrix

- If addindicatorTrue

addindicator boolean optional defaultFalse If True a MissingIndicator transform

will stack onto output of the imputer’s transform This allows a predictive estimator to

account for missingness despite imputation If a feature has no missing values at fittrain

619sklearnimpute Impute 1843

scikitlearn user guide Release 0213

time the feature won't appear on the missing indicator even if there are missing values at transformtest time

Attributes

statistics array of shape nfeatures The imputation fill value for each feature

indicator sklearnimputeMissingIndicator Indicator used to add binary indi

cators for missing values None if addindicator is False

See also

IterativeImputer Multivariate imputation of missing values

Notes

Columns which only contained missing values at fit are discarded upon transform if strategy is not "constant"

Examples

```
import numpy as np
from sklearnimpute import SimpleImputer
impmean = SimpleImputer(missing_values=np.nan, strategy='mean')
impmean.fit(7 2 3 4 np.nan 6 10 5 9)
```

SimpleImputer(add_indicator=False, copy=True, fill_value=None,

missing_values=np.nan, strategy='mean', verbose=0)

X = np.array([[2, 3, 4, np.nan, 6, 10, np.nan, 9],

print(impmean.transform(X))

```
7 2 3
4 35 6
10 35 9
```

Methods

fit(self, X, y) Fit the imputer on X

fit_transform(self, X, y) Fit to data then transform it

get_params(self, deep=True) Get parameters for this estimator

set_params(self, **params) Set the parameters of this estimator

transform(self, X) Impute all missing values in X

init(self, missing_values=np.nan, strategy='mean', fill_value=None, verbose=0, copy=True,

add_indicator=False)

fit(self, X, y=None)

Fit the imputer on X

Parameters

X array-like sparse matrix shape (n_samples, n_features) Input data where

n_samples is the number of samples and n_features is the number of features

1844 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

self SimpleImputer

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Impute all missing values in X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data to complete

Examples using sklearnimputeSimpleImputer

•Column Transformer with Mixed Types

•Imputing missing values with variants of IterativeImputer

•Imputing missing values before building an estimator

619sklearnimpute Impute 1845

scikitlearn user guide Release 0213

6192sklearnimpute IterativeImputer

classsklearnimpute IterativeImputer estimatorNone missingvaluesnan sam

pleposteriorFalse maxiter10 tol0001

nnearestfeaturesNone initialstrategy'mean'

imputationorder'ascending' minvalueNone

maxvalueNone verbose0 randomstateNone

addindicatorFalse

Multivariate imputer that estimates each feature from all the others

A strategy for imputing missing values by modeling each feature with missing values as a function of other features in a roundrobin fashion

Read more in the User Guide

Note This estimator is still experimental for now the predictions and the API (Application Programming Interface) might change without any deprecation cycle To use it you need to explicitly import enableiterativeimputer

explicitly require this experimental feature

from sklearnexperimental import enableiterativeimputer noqa

now you can import normally from sklearnimpute

from sklearnimpute import IterativeImputer

Parameters

estimator estimator object defaultBayesianRidge The estimator to use at each step of the roundrobin imputation If sampleposterior is True the estimator must support

returnstd in itspredict method

missingvalues int npnan optional defaultnpnan The placeholder for the missing values

All occurrences of missingvalues will be imputed

sampleposterior boolean defaultFalse Whether to sample from the Gaussian predictive posterior of the fitted estimator for each imputation Estimator must support returnstd

in itspredict method if set to True Set toTrue if usingIterativeImputer for

multiple imputations

maxiter int optional default10 Maximum number of imputation rounds to perform

before returning the imputations computed during the final round A round is a sin

gle imputation of each feature with missing values The stopping criterion is met

onceabsmaxXt Xt1absmaxXknownvals tol where

Xt isXat iteration t Note that early stopping is only applied if

sampleposteriorFalse

tolfloat optional default1e3 Tolerance of the stopping condition

nnearestfeatures int optional defaultNone Number of other features to use to estimate the missing values of each feature column Nearness between features is measured using

the absolute correlation coefficient between each feature pair after initial imputation To ensure coverage of features throughout the imputation process the neighbor features are

not necessarily nearest but are drawn with probability proportional to correlation for each imputed target feature Can provide significant speedup when the number of features is

huge IfNone all features will be used

initialstrategy str optional default"mean" Which strategy to use to initialize the missing values Same as the strategy parameter in sklearnimputeSimpleImputer

Valid values "mean" "median" "mostfrequent" or "constant"

1846 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

imputationorder str optional default"ascending" The order in which the features will be imputed Possible values

"ascending" From features with fewest missing values to most

"descending" From features with most missing values to fewest

"roman" Left to right

"arabic" Right to left

"random" A random order for each round

minvalue float optional defaultNone Minimum possible imputed value Default of None will set minimum to negative infinity

maxvalue float optional defaultNone Maximum possible imputed value Default of None will set maximum to positive infinity

verbose int optional default0 Verbosity flag controls the debug messages that are issued as functions are evaluated The higher the more verbose Can be 0 1 or 2

randomstate int RandomState instance or None optional defaultNone The seed of the pseudo random number generator to use Randomizes selection of estimator features if nnearestfeatures is not None the imputationorder ifrandom and the sampling from posterior if sampleposterior is True Use an integer for determinism See the Glossary

addindicator boolean optional defaultFalse If True a MissingIndicator transform will stack onto output of the imputer's transform This allows a predictive estimator to account for missingness despite imputation If a feature has no missing values at fittrain time the feature won't appear on the missing indicator even if there are missing values at transformtest time

Attributes

initialimputer object of type sklearnimputeSimpleImputer Imputer used to initialize the missing values

imputationsequence list of tuples Each tuple has featidx

neighborfeatidx estimator wherefeatidx is the current feature

to be imputed neighborfeatidx is the array of other features used to impute the current feature and estimator is the trained estimator used for the imputation Length isselfnfeatureswithmissing selfniter

niter int Number of iteration rounds that occurred Will be less than selfmaxiter if early stopping criterion was reached

nfeatureswithmissing int Number of features with missing values

indicator sklearnimputeMissingIndicator Indicator used to add binary indicators for missing values None if addindicator is False

See also

SimpleImputer Univariate imputation of missing values

Notes

To support imputation in inductive mode we store each feature's estimator during the fit phase and predict without refitting in order during the transform phase

619sklearnimpute Impute 1847

scikitlearn user guide Release 0213

Features which contain all missing values at fit are discarded upon transform

Features with missing values during transform which did not have any missing values during fit will be imputed with the initial imputation method only

References

Rcd31b817a31e1 Rcd31b817a31e2

Methods

fitself X y Fits the imputer on X and return self

fittransform self X y Fits the imputer on X and return the transformed X

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X Imputes all missing values in X

init selfestimatorNone missingvaluesnan sampleposteriorFalse maxiter10

tol0001 nnearestfeaturesNone initialstrategy'mean' imputa

tionorder'ascending' minvalueNone maxvalueNone verbose0 ran

domstateNone addindicatorFalse

fitselfXyNone

Fits the imputer on X and return self

Parameters

Xarraylike shape nsamples nfeatures Input data where “nsamples” is the number of samples and “nfeatures” is the number of features

yignored

Returns

self object Returns self

fittransform selfXyNone

Fits the imputer on X and return the transformed X

Parameters

Xarraylike shape nsamples nfeatures Input data where “nsamples” is the number of samples and “nfeatures” is the number of features

yignored

Returns

Xtarraylike shape nsamples nfeatures The imputed input data

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

1848 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Imputes all missing values in X

Note that this is stochastic and that if randomstate is not fixed repeated calls or permuted input will yield different results

Parameters

Xarraylike shape nsamples nfeatures The input data to complete

Returns

Xarraylike shape nsamples nfeatures The imputed input data

Examples using sklearnimputeliterativeImputer

- Imputing missing values with variants of IterativeImputer
- Imputing missing values before building an estimator

6193sklearnimpute MissingIndicator

classsklearnimpute MissingIndicator missingvaluesnan features'missingonly'

sparse'auto' erroronnewTrue

Binary indicators for missing values

Note that this component typically should not be used in a vanilla Pipeline consisting of transformers and a classifier but rather could be added using a FeatureUnion orColumnTransformer

Read more in the User Guide

Parameters

missingvalues number string npnan default or None The placeholder for the missing

values All occurrences of missingvalues will be indicated True in the output array the other values will be marked as False

features str optional Whether the imputer mask should represent all or a subset of features

- If "missingonly" default the imputer mask will only represent features containing

missing values during fit time

- If "all" the imputer mask will represent all features

sparse boolean or "auto" optional Whether the imputer mask format should be sparse or dense

- If "auto" default the imputer mask will be of same type as input

- If True the imputer mask will be a sparse matrix

619sklearnimpute Impute 1849

scikitlearn user guide Release 0213

• If False the imputer mask will be a numpy array
erroronnew boolean optional If True default transform will raise an error when there
are features with missing values in transform that have no missing values in fit This is
applicable only when featuresmissingonly

Attributes

features ndarray shape nmissingfeatures or nfeatures The features indices which
will be returned when calling transform They are computed during fit For
featuresall it is torangenfeatures

Examples

```
import numpy as np
from sklearn.impute import MissingIndicator
X1 = np.array(np.nan, 1, 3)
      4, 0, np.nan
      8, 1, 0
X2 = np.array(5, 1, np.nan,
              np.nan, 2, 3,
              2, 4, 0)
indicator = MissingIndicator
indicator.fit(X1)
MissingIndicator(erroronnew=True, featuresmissingonly=
missingvaluesnan, sparseauto=
X2tr = indicator.transform(X2)
X2tr
array(False, True,
       True, False,
       False, False)
```

Methods

fitself X y Fit the transformer on X
fittransform self X y Generate missing values indicator for X
getparams self deep Get parameters for this estimator
setparams self params Set the parameters of this estimator
transform self X Generate missing values indicator for X
init selfmissingvaluesnan features'missingonly' sparse'auto' erroronnewTrue
fitselfXyNone
Fit the transformer on X
Parameters
Xarraylike sparse matrix shape nsamples nfeatures Input data where
nsamples is the number of samples and nfeatures is the number of features
Returns
self object Returns self
fittransform selfXyNone
Generate missing values indicator for X

1850 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data to complete

Returns

Xndarray or sparse matrix shape nsamples nfeatures The missing indicator for

input data The data type of Xtwill be boolean

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Generate missing values indicator for X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data to complete

Returns

Xndarray or sparse matrix shape nsamples nfeatures The missing indicator for

input data The data type of Xtwill be boolean

Examples using sklearnimputeMissingIndicator

•Imputing missing values before building an estimator

620sklearnkernelapproximation Kernel Approximation

The sklearnkernelapproximation module implements several approximate kernel feature maps base on

Fourier transforms

User guide See the Kernel Approximation section for further details

kernelapproximation

AdditiveChi2Sampler Approximate feature map for additive chi2 kernel

kernelapproximationNystroem kernel Approximate a kernel map using a subset of the training

data

Continued on next page

620sklearnkernelapproximation Kernel Approximation 1851

scikitlearn user guide Release 0213

Table 6141 – continued from previous page

kernelapproximationRBFSampler gamma

Approximates feature map of an RBF (Radial Basis Function) kernel by Monte Carlo approximation of its Fourier transform

kernelapproximation

SkewedChi2Sampler Approximates feature map of the “skewed chisquared”

kernel by Monte Carlo approximation of its Fourier trans

form

6201sklearnkernelapproximation AdditiveChi2Sampler

classsklearnkernelapproximation AdditiveChi2Sampler samplesteps2 sam

pleintervalNone

Approximate feature map for additive chi2 kernel

Uses sampling the fourier transform of the kernel characteristic at regular intervals

Since the kernel that is to be approximated is additive the components of the input vectors can be treated separately Each entry in the original space is transformed into 2samplesteps1 features where samplesteps

is a parameter of the method Typical values of samplesteps include 1 2 and 3

Optimal choices for the sampling interval for certain data ranges can be computed see the reference The

default values should be reasonable

Read more in the User Guide

Parameters

samplesteps int optional Gives the number of complex sampling points

sampleinterval float optional Sampling interval Must be specified when samplesteps not

in 123

See also

SkewedChi2Sampler A Fourierapproximation to a nonadditive variant of the chi squared kernel

sklearnmetricspairwisechi2kernel The exact chi squared kernel

sklearnmetricspairwiseadditivechi2kernel The exact additive chi squared kernel

Notes

This estimator approximates a slightly different version of the additive chi squared kernel then metric

additivechi2 computes

References

See “Efficient additive kernels via explicit feature maps” A Vedaldi and A Zisserman Pattern Analysis and

Machine Intelligence 2011

Examples

from sklearn.datasets import loaddigits

from sklearn.linear_model import SGDClassifier

from sklearn.kernel_approximation import AdditiveChi2Sampler

X y loaddigitsreturnXy True

1852 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
chi2sampler AdditiveChi2Samplersamplesteps2
Xtransformed chi2samplerfittransformX y
clf SGDClassifiermaxiter5 randomstate0 tol1e3
clffitXtransformed y
SGDClassifieralpha00001 averageFalse classweightNone
earlystoppingFalse epsilon01 eta000 fitinterceptTrue
l1ratio015 learningrateoptimal loss hinge maxiter5
niter nochange5 njobsNone penaltyl2 powert05
randomstate0 shuffleTrue tol0001 validationfraction01
verbose0 warmstartFalse
clfscoreXtransformed y
09499
Methods
fitself X y Set the parameters
fittransform self X y Fit to data then transform it
getparams self deep Get parameters for this estimator
setparams self params Set the parameters of this estimator
transform self X Apply approximate feature map to X
init selfsamplesteps2 sampleintervalNone
fitselfXyNone
Set the parameters
Parameters
Xarraylike shape nsamples nfeatures Training data where nsamples is the number
of samples and nfeatures is the number of features
Returns
self object Returns the transformer
fittransform selfXyNone fitparams
Fit to data then transform it
Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X
Parameters
Xnumpy array of shape nsamples nfeatures Training set
ynumpy array of shape nsamples Target values
Returns
Xnew numpy array of shape nsamples nfeaturesnew Transformed array
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
Returns
620sklearnkernelapproximation Kernel Approximation 1853

scikitlearn user guide Release 0213

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Apply approximate feature map to X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures

Returns

Xnew array sparse matrix shape nsamples nfeatures 2samplesteps 1

Whether the return value is an array of sparse matrix depends on the type of the input X

6202sklearnkernelapproximation Nystroem

classsklearnkernelapproximation Nystroem kernel'rbf' gammaNone coef0None

degreeNone kernelparamsNone

ncomponents100 randomstateNone

Approximate a kernel map using a subset of the training data

Constructs an approximate feature map for an arbitrary kernel using a subset of the data as basis

Read more in the User Guide

Parameters

kernel string or callable default"rbf" Kernel map to be approximated A callable should

accept two arguments and the keyword arguments passed to this object as kernelparams

and should return a floating point number

gamma float defaultNone Gamma parameter for the RBF (Radial Basis Function) laplacian polynomial exponen

tial chi2 and sigmoid kernels Interpretation of the default value is left to the kernel see the

documentation for sklearnmetricspairwise Ignored by other kernels

coef0 float defaultNone Zero coefficient for polynomial and sigmoid kernels Ignored by

other kernels

degree float defaultNone Degree of the polynomial kernel Ignored by other kernels

kernelparams mapping of string to any optional Additional parameters keyword argu

ments for kernel function passed as callable object

ncomponents int Number of features to construct How many data points will be used to

construct the mapping

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

Attributes

1854 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

components array shape ncomponents nfeatures Subset of training points used to construct the feature map

componentindices array shape ncomponents Indices of components in the training set

normalization array shape ncomponents ncomponents Normalization matrix needed for embedding Square root of the kernel matrix on components

See also

RBFSampler An approximation to the RBF (Radial Basis Function) kernel using random Fourier features

sklearnmetricspairwisekernelmetrics List of builtin kernels

References

- Williams CKI and Seeger M “Using the Nystroem method to speed up kernel machines” Advances in neural information processing systems 2001
- T Yang Y Li M Mahdavi R Jin and Z Zhou “Nystroem Method vs Random Fourier Features A Theoretical and Empirical Comparison” Advances in Neural Information Processing Systems 2012

Examples

```
from sklearn import datasets svm
from sklearnkernelapproximation import Nystroem
digits datasetsloaddigitsnclass9
data digitsdata 16
clf svmLinearSVC
featuremapnystroem Nystroemgamma2
randomstate1
ncomponents300
datatransformed featuremapnystroemfittransformdata
clffitdatatransformed digitstarget
```

LinearSVCC10 classweightNone dualTrue fitinterceptTrue
interceptscaling1 losssquaredhinge maxiter1000
multiclassovr penaltyl2 randomstateNone tol00001
verbose0
clfscoredatatransformed digitstarget
09987

Methods

```
fitself X y Fit estimator to data
fittransform self X y Fit to data then transform it
getparams self deep Get parameters for this estimator
setparams self params Set the parameters of this estimator
transform self X Apply feature map to X
init selfkernel'rbf' gammaNone coef0None degreeNone kernelparamsNone
ncomponents100 randomstateNone
620sklearnkernelapproximation Kernel Approximation 1855
```

scikitlearn user guide Release 0213

fitselfXyNone

Fit estimator to data

Samples a subset of training points computes kernel on these and computes normalization matrix

Parameters

Xarraylike shapensamples nfeature Training data

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Apply feature map to X

Computes an approximate feature map using the kernel between some training points and X

Parameters

Xarraylike shapensamples nfeatures Data to transform

Returns

Xtransformed array shapensamples ncomponents Transformed data

Examples using sklearnkernelapproximationNystroem

•Explicit feature map approximation for RBF (Radial Basis Function) kernels

1856 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

6203sklearnkernelapproximation RBFSampler

classsklearnkernelapproximation RBFSampler gamma10 ncomponents100 ran

domstateNone

Approximates feature map of an RBF (Radial Basis Function) kernel by Monte Carlo approximation of its Fourier transform

It implements a variant of Random Kitchen Sinks1

Read more in the User Guide

Parameters

gamma float Parameter of RBF (Radial Basis Function) kernel expgamma x2

ncomponents int Number of Monte Carlo samples per original feature Equals the dimen
sionality of the computed feature space

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

Notes

See “Random Features for LargeScale Kernel Machines” by A Rahimi and Benjamin Recht

1 “Weighted Sums of Random Kitchen Sinks Replacing minimization with randomization in learning” by A

Rahimi and Benjamin Recht <https://people.eecs.berkeley.edu/brecht/papers/08rahreknips.pdf>

Examples

```
from sklearn.kernelapproximation import RBFSampler
```

```
from sklearn.linear_model import SGDClassifier
```

```
X = [[0, 0, 1, 1, 1, 0, 0, 1],
```

```
     [0, 0, 1, 1, 1, 0, 0, 1]]
```

```
rbf = RBFSampler(gamma=1, random_state=1)
```

```
X = rbf.fit_transform(X)
```

```
clf = SGDClassifier(max_iter=5, tol=1e-3)
```

```
clf.fit(X, y)
```

SGDClassifier(alpha=0.0001, average=False, class_weight=None,

early_stopping=False, epsilon=0.1, eta=0.001, fit_intercept=True,

l1_ratio=0.15, learning_rate='optimal', loss='hinge', max_iter=5,

min_iter=5, n_iter_no_change=5, n_jobs=None, penalty='l2', power_t=0.5,

random_state=None, shuffle=True, tol=0.0001, validation_fraction=0.1,

verbose=0, warm_start=False,

clf.score(X, y)

10

Methods

fit(self, X, y) Fit the model with X

fit_transform(self, X, y) Fit to data then transform it

Continued on next page

620sklearnkernelapproximation Kernel Approximation 1857

scikitlearn user guide Release 0213

Table 6144 - continued from previous page

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X Apply the approximate feature map to X

init selfgamma10 ncomponents100 randomstateNone

fitselfXyNone

Fit the model with X

Samples random projection according to nfeatures

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data where

nsamples in the number of samples and nfeatures is the number of features

Returns

self object Returns the transformer

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Apply the approximate feature map to X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures New data where nsamples

in the number of samples and nfeatures is the number of features

1858 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

Xnew arraylike shape nsamples ncomponents

Examples using sklearnkernelapproximationRBFSampler

•Explicit feature map approximation for RBF (Radial Basis Function) kernels

6204sklearnkernelapproximation SkewedChi2Sampler

classssklearnkernelapproximation SkewedChi2Sampler skewedness10

ncomponents100 ran

domstateNone

Approximates feature map of the “skewed chisquared” kernel by Monte Carlo approximation of its Fourier transform

Read more in the User Guide

Parameters

skewedness float “skewedness” parameter of the kernel Needs to be crossvalidated

ncomponents int number of Monte Carlo samples per original feature Equals the dimensionality of the computed feature space

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

See also

AdditiveChi2Sampler A different approach for approximating an additive variant of the chi squared kernel

sklearnmetricspairwisechi2kernel The exact chi squared kernel

References

See “Random Fourier Approximations for Skewed Multiplicative Histogram Kernels” by Fuxin Li Catalin Ionescu and Cristian Sminchisescu

Examples

from sklearnkernelapproximation import SkewedChi2Sampler

from sklearnlinearmodel import SGDClassifier

X 0 0 1 1 1 0 0 1

y 0 0 1 1

chi2feature SkewedChi2Samplerskewedness01

ncomponents10

randomstate0

Xfeatures chi2featurefittransformX y

clf SGDClassifiermaxiter10 tol1e3

clffitXfeatures y

SGDClassifieralpha00001 averageFalse classweightNone

620sklearnkernelapproximation Kernel Approximation 1859

scikitlearn user guide Release 0213
earlystoppingFalse epsilon01 eta000 fitinterceptTrue
l1ratio015 learningrateoptimal loss hinge maxiter10
niter nochange5 njobsNone penaltyl2 powert05
randomstateNone shuffleTrue tol0001 validationfraction01
verbose0 warmstartFalse
clf.score(X, features, y)
10
Methods
fitself X y Fit the model with X
fittransform self X y Fit to data then transform it
getparams self deep Get parameters for this estimator
setparams self params Set the parameters of this estimator
transform self X Apply the approximate feature map to X
init selfskewedness10 ncomponents100 randomstateNone
fitselfXyNone
Fit the model with X
Samples random projection according to nfeatures
Parameters
Xarraylike shape nsamples nfeatures Training data where nsamples is the number
of samples and nfeatures is the number of features
Returns
self object Returns the transformer
fittransform selfXyNone fitparams
Fit to data then transform it
Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X
Parameters
Xnumpy array of shape nsamples nfeatures Training set
ynumpy array of shape nsamples Target values
Returns
Xnew numpy array of shape nsamples nfeaturesnew Transformed array
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values
1860 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

transform selfX

Apply the approximate feature map to X

Parameters

Xarraylike shape nsamples nfeatures New data where nsamples in the number of samples and nfeatures is the number of features All values of X must be strictly greater than “skewedness”

Returns

Xnew arraylike shape nsamples ncomponents

621sklearnkernelridge Kernel Ridge Regression

Modulesklearnkernelridge implements kernel ridge regression

User guide See the Kernel ridge regression section for further details

kernelridgeKernelRidge alpha kernel Kernel ridge regression

6211sklearnkernelridge KernelRidge

classssklearnkernelridge KernelRidge alpha1 kernel’linear’ gammaNone degree3

coef01 kernelparamsNone

Kernel ridge regression

Kernel ridge regression KRR combines ridge regression linear least squares with l2norm regularization with the kernel trick It thus learns a linear function in the space induced by the respective kernel and the data For nonlinear kernels this corresponds to a nonlinear function in the original space

The form of the model learned by KRR is identical to support vector regression SVR (Support Vector Regression) However different loss functions are used KRR uses squared error loss while support vector regression uses epsiloninsensitive loss

both combined with l2 regularization In contrast to SVR (Support Vector Regression) fitting a KRR model can be done in closedform and is typically faster for mediumsized datasets On the other hand the learned model is nonsparse and thus slower

than SVR (Support Vector Regression) which learns a sparse model for epsilon 0 at predictiontime

This estimator has builtin support for multivariate regression ie when y is a 2darray of shape nsamples

ntargets

Read more in the User Guide

Parameters

alpha float arraylike shape ntargets Small positive values of alpha improve the conditioning of the problem and reduce the variance of the estimates Alpha corresponds to 2C1 in other linear models such as LogisticRegression or LinearSVC If an array is 621sklearnkernelridge Kernel Ridge Regression 1861

scikitlearn user guide Release 0213

passed penalties are assumed to be specific to the targets Hence they must correspond in number

kernel string or callable default"linear" Kernel mapping used internally A callable should accept two arguments and the keyword arguments passed to this object as kernelparams and should return a floating point number Set to "precomputed" in order to pass a precomputed kernel matrix to the estimator methods instead of samples

gamma float defaultNone Gamma parameter for the RBF (Radial Basis Function) laplacian polynomial exponential chi2 and sigmoid kernels Interpretation of the default value is left to the kernel see the documentation for sklearnmetricspairwise Ignored by other kernels

degree float default3 Degree of the polynomial kernel Ignored by other kernels

coef0 float default1 Zero coefficient for polynomial and sigmoid kernels Ignored by other kernels

kernelparams mapping of string to any optional Additional parameters keyword arguments for kernel function passed as callable object

Attributes

dualcoef array shape nsamples or nsamples ntargets Representation of weight vectors in kernel space

Xfit arraylike sparse matrix shape nsamples nfeatures Training data which is also required for prediction If kernel "precomputed" this is instead the precomputed training matrix shape nsamples nsamples

See also

sklearnlinearmodelRidge Linear ridge regression

sklearnsvmSVR Support Vector Regression implemented using libsvm

References

• Kevin P Murphy "Machine Learning A Probabilistic Perspective" The MIT Press chapter 1443 pp 492493

Examples

```
from sklearnkernelridge import KernelRidge
```

```
import numpy as np
```

```
nsamples nfeatures 10 5
```

```
rng np.random.RandomState0
```

```
y rng.randnnsamples
```

```
X rng.randnnsamples nfeatures
```

```
clf KernelRidgealpha10
```

```
clffitX y
```

```
KernelRidgealpha10 coef01 degree3 gammaNone kernellinear
```

```
kernelparamsNone
```

Methods

1862 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

fitself X y sampleweight Fit Kernel Ridge regression model

getparams self deep Get parameters for this estimator

predict self X Predict using the kernel ridge model

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfalpha1 kernel'linear' gammaNone degree3 coef01 kernelparamsNone

fitselfXyNone sampleweightNone

Fit Kernel Ridge regression model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data If kernel

precomputed" this is instead a precomputed kernel matrix shape nsamples nsamples

yarraylike shape nsamples or nsamples ntargets Target values

sampleweight float or arraylike of shape nsamples Individual weights for each sample ignored if None is passed

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the kernel ridge model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Samples If kernel

precomputed" this is instead a precomputed kernel matrix shape nsamples

nsamplesfitted where nsamplesfitted is the number of samples used in the fitting for this estimator

Returns

Carray shape nsamples or nsamples ntargets Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as 1 - uv where u is the residual sum of squares ytrue ypred

2sum and v is the total sum of squares ytrue ytruemean 2sum The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

621sklearnkernelridge Kernel Ridge Regression 1863

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures Test samples For some estimators this may be a precomputed kernel matrix instead shape nsamples nsamplesfitted where nsamplesfitted is the number of samples used in the fitting for the estimator yarraylike shape nsamples or nsamples noutputs True values for X sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

Examples using sklearnkernelridgeKernelRidge

- Comparison of kernel ridge regression and SVR (Support Vector Regression)
- Comparison of kernel ridge and Gaussian process regression

622sklearnlinearmodel Generalized Linear Models

The sklearnlinearmodel module implements generalized linear models It includes Ridge regression Bayesian Regression Lasso and Elastic Net estimators computed with Least Angle Regression and coordinate descent It also implements Stochastic Gradient Descent related algorithms

User guide See the Generalized Linear Models section for further details

linearmodelARDRegression niter tol Bayesian ARD regression

linearmodelBayesianRidge niter tol Bayesian ridge regression

linearmodelElasticNet alpha l1ratio Linear regression with combined L1 and L2 priors as regularizer

linearmodelElasticNetCV l1ratio eps Elastic Net model with iterative fitting along a regularization path

linearmodelHuberRegressor epsilon Linear regression model that is robust to outliers

linearmodelLars fitintercept verbose Least Angle Regression model aka

Continued on next page

1864 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Table 6148 - continued from previous page

linearmodelLarsCV fitintercept Crossvalidated Least Angle Regression model

linearmodelLasso alpha fitintercept Linear Model trained with L1 prior as regularizer aka the Lasso

linearmodelLassoCV eps nalphas Lasso linear model with iterative fitting along a regularization path

linearmodelLassoLars alpha Lasso model fit with Least Angle Regression aka

linearmodelLassoLarsCV fitintercept Crossvalidated Lasso using the LARS algorithm

linearmodelLassoLarsIC criterion Lasso model fit with Lars using BIC or AIC for model selection

linearmodelLinearRegression Ordinary least squares Linear Regression

linearmodelLogisticRegression penalty Logistic Regression aka logit MaxEnt classifier

linearmodelLogisticRegressionCV Cs Logistic Regression CV (Cross-Validation) aka logit MaxEnt classifier

linearmodelMultiTaskLasso alpha Multitask Lasso model trained with L1L2 mixednorm as regularizer

linearmodelMultiTaskElasticNet alpha Multitask ElasticNet model trained with L1L2 mixed norm as regularizer

linearmodelMultiTaskLassoCV eps Multitask Lasso model trained with L1L2 mixednorm as regularizer

linearmodelMultiTaskElasticNetCV Multitask L1L2 ElasticNet with builtin crossvalidation

linearmodelOrthogonalMatchingPursuit Orthogonal Matching Pursuit model OMP

linearmodelOrthogonalMatchingPursuitCV Crossvalidated Orthogonal Matching Pursuit model OMP

linearmodelPassiveAggressiveClassifier Passive Aggressive Classifier

linearmodelPassiveAggressiveRegressor C Passive Aggressive Regressor

linearmodelPerceptron penalty alpha Read more in the User Guide

linearmodelRANSACRegressor RANSAC RANDOM Sample Consensus algorithm

linearmodelRidge alpha fitintercept Linear least squares with l2 regularization

linearmodelRidgeClassifier alpha Classifier using Ridge regression

linearmodelRidgeClassifierCV alphas Ridge classifier with builtin crossvalidation

linearmodelRidgeCV alphas Ridge regression with builtin crossvalidation

linearmodelSGDClassifier loss penalty Linear classifiers SVM (Support Vector Machine) logistic regression also with SGD training

linearmodelSGDRegressor loss penalty Linear model fitted by minimizing a regularized empirical loss with SGD

linearmodelTheilSenRegressor TheilSen Estimator robust multivariate regression model

6221sklearnlinearmodel ARDRegression

classsklearnlinearmodel ARDRegression niter300 tol0.001 alpha1e06

06 alpha21e06 lambda11e06

lambda21e06 computescoreFalse thresh

oldlambda100000 fitinterceptTrue normal

izeFalse copyXTrue verboseFalse

Bayesian ARD regression

Fit the weights of a regression model using an ARD prior The weights of the regression model are assumed to be in Gaussian distributions Also estimate the parameters lambda precisions of the distributions of the weights and alpha precision of the distribution of the noise The estimation is done by an iterative procedure

622sklearnlinearmodel Generalized Linear Models 1865

scikitlearn user guide Release 0213

Evidence Maximization

Read more in the User Guide

Parameters

niter int optional Maximum number of iterations Default is 300

tolfloat optional Stop the algorithm if w has converged Default is 1e3

alpha1 float optional Hyperparameter shape parameter for the Gamma distribution prior over the alpha parameter Default is 1e6

alpha2 float optional Hyperparameter inverse scale parameter rate parameter for the Gamma distribution prior over the alpha parameter Default is 1e6

lambda1 float optional Hyperparameter shape parameter for the Gamma distribution prior over the lambda parameter Default is 1e6

lambda2 float optional Hyperparameter inverse scale parameter rate parameter for the Gamma distribution prior over the lambda parameter Default is 1e6

computescore boolean optional If True compute the objective function at each step of the model Default is False

thresholdlambda float optional threshold for removing pruning weights with high precision from the computation Default is 1e4

fitintercept boolean optional whether to calculate the intercept for this model If set to false no intercept will be used in calculations eg data is expected to be already centered Default is True

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized before regression by subtracting the mean and dividing by the l2norm If you wish to standardize please use sklearnpreprocessingStandardScaler before callingfit on an estimator with normalizeFalse

copyX boolean optional default True If True X will be copied else it may be overwritten

verbose boolean optional default False Verbose mode when fitting the model

Attributes

coef array shape nfeatures Coefficients of the regression model mean of distribution

alpha float estimated precision of the noise

lambda array shape nfeatures estimated precisions of the weights

sigma array shape nfeatures nfeatures estimated variancecovariance matrix of the weights

scores float if computed value of the objective function to be maximized

Notes

For an example see exampleslinearmodelplotardpy

1866 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

References

D J C MacKay Bayesian nonlinear modeling for the prediction competition ASHRAE Transactions 1994
R Salakhutdinov Lecture notes on Statistical Machine Learning <http://www.utstat.toronto.edu/~dursalakhu/sta4273/notes/Lecture2.pdf> page 15 Their β is our self.alpha Their α is our self.lambda
ARD is a little different than the slide only dimensions/features for which self.lambda self.threshold lambda are kept and the rest are discarded

Examples

```
from sklearn import linear_model
clf = linear_model.ARDRegression
clf.fit(0 1 1 2 2 0 1 2
```

```
ARDRegression(alpha=1e-06, alpha2=1e-06, compute_score=False,
copy_X=True, fit_intercept=True, lambda1=1e-06, lambda2=1e-06,
n_iter=300, normalize=False, threshold_lambda=100000, tol=0.001,
verbose=False)
clf.predict(1 1
array(1
```

Methods

`fit(self, X, y)` Fit the ARDRegression model according to the given training data and parameters
`get_params(self, deep=True)` Get parameters for this estimator
`predict(self, X, return_std=False)` Predict using the linear model
`score(self, X, y, sample_weight=None)` Returns the coefficient of determination R^2 of the prediction
`set_params(self, **params)` Set the parameters of this estimator
`init(self, n_iter=300, tol=0.001, alpha1=1e-06, alpha2=1e-06, lambda1=1e-06, lambda2=1e-06, compute_score=False, threshold_lambda=100000, fit_intercept=True, normalize=False, copy_X=True, verbose=False)`
`fit(self, X, y)`

Fit the ARDRegression model according to the given training data and parameters
Iterative procedure to maximize the evidence

Parameters

`X` array-like shape (n_samples, n_features) Training vector where n_samples is the number of samples and n_features is the number of features
`y` array shape (n_samples,) Target values integers Will be cast to X's dtype if necessary
Returns

`self` returns an instance of self
`get_params(self, deep=True)`
Get parameters for this estimator
Parameters

622sklearn.linear_model.Generalized Linear Models 1867

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXreturnstdFalse

Predict using the linear model

In addition to the mean of the predictive distribution also its standard deviation can be returned

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Samples

returnstd boolean optional Whether to return the standard deviation of posterior prediction

Returns

ymean array shape nsamples Mean of predictive distribution of query points

ystd array shape nsamples Standard deviation of predictive distribution of query points

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

1868 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns
self

Examples using sklearnlinearmodelARDRegression

•Automatic Relevance Determination Regression ARD

6222sklearnlinearmodel BayesianRidge

classssklearnlinearmodel BayesianRidge niter300 tol0001 alpha11e06

alpha21e06 lambda11e06 lambda21e

06 computescoreFalse fitinterceptTrue

normalizeFalse copyXTrue verboseFalse

Bayesian ridge regression

Fit a Bayesian ridge model See the Notes section for details on this implementation and the optimization of the regularization parameters lambda precision of the weights and alpha precision of the noise

Read more in the User Guide

Parameters

niter int optional Maximum number of iterations Default is 300 Should be greater than or equal to 1

tolfloat optional Stop the algorithm if w has converged Default is 1e3

alpha1 float optional Hyperparameter shape parameter for the Gamma distribution prior over the alpha parameter Default is 1e6

alpha2 float optional Hyperparameter inverse scale parameter rate parameter for the Gamma distribution prior over the alpha parameter Default is 1e6

lambda1 float optional Hyperparameter shape parameter for the Gamma distribution prior over the lambda parameter Default is 1e6

lambda2 float optional Hyperparameter inverse scale parameter rate parameter for the Gamma distribution prior over the lambda parameter Default is 1e6

computescore boolean optional If True compute the log marginal likelihood at each iteration of the optimization Default is False

fitintercept boolean optional default True Whether to calculate the intercept for this model The intercept is not treated as a probabilistic parameter and thus has no associated variance

If set to False no intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when fitintercept is set to False If True the regressors X will be normalized before regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before callingfit on an estimator with normalizeFalse

copyX boolean optional default True If True X will be copied else it may be overwritten

verbose boolean optional default False Verbose mode when fitting the model

Attributes

coef array shape nfeatures Coefficients of the regression model mean of distribution

622sklearnlinearmodel Generalized Linear Models 1869

scikitlearn user guide Release 0213

intercept float Independent term in decision function Set to 00 if fitintercept
False

alpha float Estimated precision of the noise

lambda float Estimated precision of the weights

sigma array shape nfeatures nfeatures Estimated variancecovariance matrix of the
weights

scores array shape niter 1 If computedscore is True value of the log marginal
likelihood to be maximized at each iteration of the optimization The array starts with the
value of the log marginal likelihood obtained for the initial values of alpha and lambda and
ends with the value obtained for the estimated alpha and lambda

niter int The actual number of iterations to reach the stopping criterion

Notes

There exist several strategies to perform Bayesian ridge regression This implementation is based on the algo
rithm described in Appendix A of Tipping 2001 where updates of the regularization parameters are done as
suggested in MacKay 1992 Note that according to A New View of Automatic Relevance Determination Wipf
and Nagarajan 2008 these update rules do not guarantee that the marginal likelihood is increasing between two
consecutive iterations of the optimization

References

D J C MacKay Bayesian Interpolation Computation and Neural Systems V ol 4 No 3 1992

M E Tipping Sparse Bayesian Learning and the Relevance Vector Machine Journal of Machine Learning
Research V ol 1 2001

Examples

```
from sklearn import linearmodel
```

```
clf = linearmodel.BayesianRidge
```

```
clffit00 1 1 2 2 0 1 2
```

```
BayesianRidge(alpha=11e06, alpha2=1e06, computedscore=False,
```

```
copyX=True, fitintercept=True, lambda=11e06, lambda2=1e06,
```

```
niter=300, normalize=False, tol=0001, verbose=False,
```

```
clf.predict(1, 1,
```

```
array(1,
```

Methods

fitself X y sampleweight Fit the model

getparams self deep Get parameters for this estimator

predict self X returnstd Predict using the linear model

score self X y sampleweight Returns the coefficient of determination R2 of the pre
diction

Continued on next page

1870 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Table 6150 – continued from previous page

setparams self params Set the parameters of this estimator

init selfniter300 tol0001 alpha11e06 alpha21e06 lambda11e

06lambda21e06 computescoreFalse fitinterceptTrue normalizeFalse

copyXTrue verboseFalse

fitselfXysampleweightNone

Fit the model

Parameters

Xnumpy array of shape nsamplesnfeatures Training data

ynumpy array of shape nsamples Target values Will be cast to X's dtype if necessary

sampleweight numpy array of shape nsamples Individual weights for each sample

New in version 020 parameter sampleweight support to BayesianRidge

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXreturnstdFalse

Predict using the linear model

In addition to the mean of the predictive distribution also its standard deviation can be returned

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Samples

returnstd boolean optional Whether to return the standard deviation of posterior predic

tion

Returns

ymean array shape nsamples Mean of predictive distribution of query points

ystd array shape nsamples Standard deviation of predictive distribution of query

points

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{true\text{mean}}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

622sklearnlinearmodel Generalized Linear Models 1871

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures Test samples For some estimators this may be a precomputed kernel matrix instead shape nsamples nsamplesfitted where nsamplesfitted is the number of samples used in the fitting for the estimator yarraylike shape nsamples or nsamples noutputs True values for X sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

Examples using sklearnlinearmodelBayesianRidge

- Feature agglomeration vs univariate selection
- Imputing missing values with variants of IterativeImputer
- Bayesian Ridge Regression

6223sklearnlinearmodel ElasticNet

classsklearnlinearmodel ElasticNet alpha10 l1ratio05 fitinterceptTrue nor malizeFalse precomputeFalse maxiter1000

copyXTrue tol00001 warmstartFalse posi

tiveFalse randomstateNone selection’cyclic’

Linear regression with combined L1 and L2 priors as regularizer

Minimizes the objective function

1 2nsamples y Xw22

alpha l1ratio w1

05alpha1 l1ratio w22

If you are interested in controlling the L1 and L2 penalty separately keep in mind that this is equivalent to

aL1 bL2

1872 Chapter 6 API (Application Programming Interface) Reference

where
 $\alpha = \frac{a}{a+b}$ and $\lambda_{ratio} = \frac{a}{a+b}$
The parameter λ_{ratio} corresponds to α in the glmnet R package while α corresponds to the λ parameter in glmnet Specifically $\lambda_{ratio} = 1$ is the lasso penalty Currently $\lambda_{ratio} = 0.01$ is not reliable unless you supply your own sequence of α
Read more in the User Guide
Parameters

α float optional Constant that multiplies the penalty terms Defaults to 10 See the notes for the exact mathematical meaning of this parameter “ $\alpha = 0$ ” is equivalent to an ordinary least square solved by the LinearRegression object For numerical reasons using $\alpha = 0$ with the Lasso object is not advised Given this you should use the LinearRegression object
 λ_{ratio} float The ElasticNet mixing parameter with 0 $\lambda_{ratio} = 1$ For $\lambda_{ratio} = 0$ the penalty is an L2 penalty For $\lambda_{ratio} = 1$ it is an L1 penalty For 0 $\lambda_{ratio} = 1$ the penalty is a combination of L1 and L2
 fit_intercept bool Whether the intercept should be estimated or not If False the data is assumed to be already centered
 normalize boolean optional default False This parameter is ignored when fit_intercept is set to False If True the regressors X will be normalized before regression by subtracting the mean and dividing by the ℓ_2 norm If you wish to standardize please use sklearn.preprocessing.StandardScaler before calling fit on an estimator with $\text{normalize} = \text{False}$
 precompute True False arraylike Whether to use a precomputed Gram matrix to speed up calculations The Gram matrix can also be passed as argument For sparse input this option is always True to preserve sparsity
 max_iter int optional The maximum number of iterations
 copy_X boolean optional default True If True X will be copied else it may be overwritten
 tol float optional The tolerance for the optimization if the updates are smaller than tol the optimization code checks the dual gap for optimality and continues until it is smaller than tol
 warm_start bool optional When set to True reuse the solution of the previous call to fit as initialization otherwise just erase the previous solution See the Glossary
 positive bool optional When set to True forces the coefficients to be positive
 random_state int RandomState instance or None optional default None The seed of the pseudo random number generator that selects a random feature to update If int random_state is the seed used by the random number generator If RandomState instance random_state is the random number generator If None the random number generator is the RandomState instance used by np.random Used when selection ‘random’
 selection str default ‘cyclic’ If set to ‘random’ a random coefficient is updated every iteration rather than looping over features sequentially by default This setting to ‘random’ often leads to significantly faster convergence especially when tol is higher than $1e4$

Attributes
 coef array shape nfeatures ntargets nfeatures parameter vector w in the cost function formula

scikitlearn user guide Release 0213

sparsecoef scipysparse matrix shape nfeatures 1 ntargets nfeatures sparse
representation of the fitted coef

intercept float array shape ntargets independent term in decision function

niter arraylike shape ntargets number of iterations run by the coordinate descent solver
to reach the specified tolerance

See also

ElasticNetCV Elastic net model with best model selection by crossvalidation

SGDRegressor implements elastic net regression with incremental training

SGDClassifier implements logistic regression with elastic net penalty

SGDClassifierlosslog penaltyelasticnet

Notes

To avoid unnecessary memory duplication the X argument of the fit method should be directly passed as a

Fortrancontiguous numpy array

Examples

```
from sklearnlinear import ElasticNet
```

```
from sklearn.datasets import make_regression
```

```
X, y = make_regression(n_features=2, random_state=0)
```

```
regr = ElasticNet(random_state=0)
```

```
regr.fit(X, y)
```

```
ElasticNet(alpha=1.0, copy_X=True, fit_intercept=True, l1_ratio=0.5,
```

```
max_iter=1000, normalize=False, positive=False, precompute=False,
```

```
random_state=0, selection_cyclic=True, tol=0.0001, warm_start=False)
```

```
print regr.coef_
```

```
1883816048 6455968825
```

```
print regr.intercept_
```

```
1451
```

```
print regr.predict(0, 0)
```

```
1451
```

Methods

fitself X y checkinput Fit model with coordinate descent

get_paramsself deep Get parameters for this estimator

path X y l1_ratio eps n_alphas Compute elastic net path with coordinate descent

predictself X Predict using the linear model

scoreself X y sample_weight Returns the coefficient of determination R2 of the pre
diction

set_paramsself params Set the parameters of this estimator

1874 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

init selfalpha10 l1ratio05 fitinterceptTrue normalizeFalse precomputeFalse

maxiter1000 copyXTrue tol00001 warmstartFalse positiveFalse ran

domstateNone selection'cyclic'

fitselfXycheckinputTrue

Fit model with coordinate descent

Parameters

Xndarray or scipysparse matrix nsamples nfeatures Data

yndarray shape nsamples or nsamples ntargets Target Will be cast to X's dtype

if necessary

checkinput boolean defaultTrue Allow to bypass several input checking Don't use

this parameter unless you know what you do

Notes

Coordinate descent is an algorithm that considers each column of data at a time hence it will automatically

convert the X input as a Fortrancontiguous numpy array if necessary

To avoid memory reallocation it is advised to allocate the initial data in memory directly using that format

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

staticpathXyl1ratio05 eps0001 nalphas100 alphasNone precompute'auto'

XyNone copyXTrue coefinitNone verboseFalse returnniterFalse posi

tiveFalse checkinputTrue params

Compute elastic net path with coordinate descent

The elastic net optimization function varies for mono and multioutputs

For monooutput tasks it is

1 2nsamples y Xw22

alpha l1ratio w1

05alpha1 l1ratio w22

For multioutput tasks it is

1 2 nsamples Y XWFro2

alpha l1ratio W21

05alpha1 l1ratio WFro2

Where

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

622sklearnlinearmodel Generalized Linear Models 1875

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran contiguous data to avoid unnecessary memory duplication If yis monooutput then X can be sparse
yndarray shape nsamples or nsamples noutputs Target values
l1ratio float optional float between 0 and 1 passed to elastic net scaling between l1 and l2 penalties l1ratio1 corresponds to the Lasso
eps float Length of the path eps1e3 means that alphamin alphamax 1e3
nalphas int optional Number of alphas along the regularization path
alphas ndarray optional List of alphas where to compute the models If None alphas are set automatically
precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to speed up calculations If set to auto let us decide The Gram matrix can also be passed as argument
Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only when the Gram matrix is precomputed
copyX boolean optional default True If True X will be copied else it may be over written
coefinit array shape nfeatures None The initial values of the coefficients
verbose bool or integer Amount of verbosity
returnniter bool whether to return the number of iterations or not
positive bool default False If set to True forces coefficients to be positive Only allowed whenyndim 1
checkinput bool default True Skip input validation checks including the Gram matrix when provided assuming there are handled by the caller when checkinputFalse
params kwargs keyword arguments passed to the coordinate descent solver
Returns
alphas array shape nalphas The alphas along the path where models are computed
coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients along the path
dualgaps array shape nalphas The dual gaps at the end of the optimization for each alpha
niters arraylike shape nalphas The number of iterations taken by the coordinate descent optimizer to reach the specified tolerance for each alpha Is returned when returnniter is set to True
See also
MultiTaskElasticNet
MultiTaskElasticNetCV
ElasticNet
ElasticNetCV
1876 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Notes

For an example see `examples/linear_model/plot_lasso_coordinated_descent_path.py`

`predict_selfX`

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape `nsamples nfeatures` Samples

Returns

Carray shape `nsamples` Returns predicted values

`score_selfXysampleweight`None

Returns the coefficient of determination R^2 of the prediction

The coefficient R^2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R^2 score of 0.0

Parameters

Xarraylike shape `nsamples nfeatures` Test samples For some estimators this may

be a precomputed kernel matrix instead shape `nsamples nsamplesfitted` where

`nsamplesfitted` is the number of samples used in the fitting for the estimator

yarraylike shape `nsamples` or `nsamples noutputs` True values for X

sampleweight arraylike shape `nsamples` optional Sample weights

Returns

score float R^2 of `self.predictX` wrt y

Notes

The R^2 score used when calling `score` on a regressor will use `multioutputuniformaverage`

from version 0.23 to keep consistent with `metricsr2score` This will influence the score

method of all the multioutput regressors except for `MultiOutputRegressor`

To specify the default value manually and avoid the warning please either call `metricsr2score`

directly or make a custom scorer with `metricsmakescorer` the builtin scorer `r2` uses

`multioutputuniformaverage`

`setparams selfparams`

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form `componentparameter` so that it's possible to update each component

of a nested object

Returns

self

`sparsecoef`

sparse representation of the fitted coef

622sklearnlinear_model Generalized Linear Models 1877

scikitlearn user guide Release 0213

Examples using sklearnlinearmodelElasticNet

- Lasso and Elastic Net for Sparse Signals

- Train error vs Test error

6224sklearnlinearmodel HuberRegressor

classsklearnlinearmodel HuberRegressor epsilon135 maxiter100 alpha00001

warmstartFalse fitinterceptTrue tol1e05

Linear regression model that is robust to outliers

The Huber Regressor optimizes the squared loss for the samples where $y - Xw \leq \sigma$

epsilon and the absolute loss for the samples where $y - Xw > \sigma$ where w

and sigma are parameters to be optimized The parameter sigma makes sure that if y is scaled up or down by a

certain factor one does not need to rescale epsilon to achieve the same robustness Note that this does not take

into account the fact that the different features of X may be of different scales

This makes sure that the loss function is not heavily influenced by the outliers while not completely ignoring

their effect

Read more in the User Guide

New in version 018

Parameters

epsilon float greater than 10 default 135 The parameter epsilon controls the number of

samples that should be classified as outliers The smaller the epsilon the more robust it is

to outliers

maxiter int default 100 Maximum number of iterations that scipyoptimizefminlbfgsb

should run for

alpha float default 00001 Regularization parameter

warmstart bool default False This is useful if the stored attributes of a previously used

model has to be reused If set to False then the coefficients will be rewritten for every call

to fit See the Glossary

fitintercept bool default True Whether or not to fit the intercept This can be set to False if

the data is already centered around the origin

tolfloat default 1e5 The iteration will stop when $\max_j |g_i| \leq 1$

ntol where g_i is the i th component of the projected gradient

Attributes

coef array shape nfeatures Features got by optimizing the Huber loss

intercept float Bias

scale float The value by which $y - Xw - c$ is scaled down

niter int Number of iterations that fminlbfgsb has run for

Changed in version 020 In SciPy 100 the number of lbfgs iterations may exceed

maxiter niter will now report at most maxiter

outliers array shape nsamples A boolean mask which is set to True where the samples

are identified as outliers

1878 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

References

Re4616ef910fb1 Re4616ef910fb2

Examples

```
import numpy as np
from sklearnlinear import HuberRegressor LinearRegression
from sklearn.datasets import make_regression
rng = np.random.RandomState(0)
X, y, coef = make_regression(
    n_samples=200, n_features=2, noise=40, coef=True, random_state=0
)
X4 = rng.uniform(10, 20, 4, 2)
y4 = rng.uniform(10, 20, 4)
huber = HuberRegressor
fitX, y = huber.fit(X, y)
huber.score(X, y)
7284608623514573
huber.predict(X1)
array(8067200)
linear = LinearRegression
fitX, y = linear.fit(X, y)
print True coefficients, coef
True coefficients: 204923 341698
print Huber coefficients, huber.coef
Huber coefficients: 177906 310106
print Linear Regression coefficients, linear.coef
Linear Regression coefficients: 19221 70226
Methods
fitself X, y, sample_weight Fit the model according to the given training data
get_params self deep Get parameters for this estimator
predict self X Predict using the linear model
score self X, y, sample_weight Returns the coefficient of determination R2 of the prediction
set_params self params Set the parameters of this estimator
init self epsilon=135 max_iter=100 alpha=0.0001 warm_start=False fit_intercept=True
tol=1e-05
fitself X, y, sample_weight=None
Fit the model according to the given training data
Parameters
X arraylike shape (n_samples, n_features) Training vector where n_samples is the number of samples and n_features is the number of features
y arraylike shape (n_samples) Target vector relative to X
sample_weight arraylike shape (n_samples) Weight given to each sample
Returns
self object
622sklearnlinear model Generalized Linear Models 1879
```

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

1880 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

self

Examples using sklearnlinearmodelHuberRegressor

•HuberRegressor vs Ridge on dataset with strong outliers

•Robust linear estimator fitting

6225sklearnlinearmodel Lars

classssklearnlinearmodel LarsfitinterceptTrue verboseFalse normalizeTrue precom

pute'auto' nnonzerocoefs500 eps2220446049250313e

16copyXTrue fitpathTrue positiveFalse

Least Angle Regression model aka LAR

Read more in the User Guide

Parameters

fitintercept boolean Whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

verbose boolean or integer optional Sets the verbosity amount

normalize boolean optional default True This parameter is ignored when fitintercept

is set to False If True the regressors X will be normalized before regression by sub

tracting the mean and dividing by the l2norm If you wish to standardize please use

sklearnpreprocessingStandardScaler before calling fit on an estimator

withnormalizeFalse

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to

speed up calculations If set to auto let us decide The Gram matrix can also be passed

as argument

nnonzerocoefs int optional Target number of nonzero coefficients Use npinf for no

limit

eps float optional The machineprecision regularization in the computation of the Cholesky

diagonal factors Increase this for very illconditioned systems Unlike the tol parameter in

some iterative optimizationbased algorithms this parameter does not control the tolerance

of the optimization

copyX boolean optional default True If True X will be copied else it may be overwritten

fitpath boolean If True the full path is stored in the coefpath attribute If you compute

the solution for a large problem or many targets setting fitpath toFalse will lead to

a speedup especially with a small alpha

positive boolean defaultFalse Restrict coefficients to be 0 Be aware that you might

want to remove fitintercept which is set True by default

Deprecated since version 020 The option is broken and deprecated It will be removed in

v022

Attributes

alphas array shape nalphas 1 list of ntargets such arrays Maximum of covari

ances in absolute value at each iteration nalphas is either nnonzerocoefs or

nfeatures whichever is smaller

622sklearnlinearmodel Generalized Linear Models 1881

scikitlearn user guide Release 0213

active list length nalphas list of ntargets such lists Indices of active variables at the end of the path

coefpath array shape nfeatures nalphas 1 list of ntargets such arrays The varying values of the coefficients along the path It is not present if the fitpath parameter is False

coef array shape nfeatures or ntargets nfeatures Parameter vector w in the formula tion formula

intercept float array shape ntargets Independent term in decision function

niter arraylike or int The number of iterations taken by larspath to find the grid of alphas for each target

See also

larspath LarsCV

sklearn.decomposition.sparse_encode

Examples

```
from sklearn import linear_model
reg = linear_model.Lars(nnonzero_coefs=1)
reg.fit(1 1 0 0 1 1 1 1 1 1 0 1 1 1 1)
```

LarscopyX=True eps=fit_intercept=True fit_path=True

nnonzero_coefs=1 normalize=True positive=False precompute='auto'

verbose=False

print_reg_coef

0 1 1

Methods

fitself X y Xy Fit the model using X y as training data

get_params self deep Get parameters for this estimator

predict self X Predict using the linear model

score self X y sample_weight Returns the coefficient of determination R2 of the pre diction

set_params self params Set the parameters of this estimator

init self fit_intercept=True verbose=False normalize=True precompute='auto'

nnonzero_coefs=500 eps=2.220446049250313e-16 copyX=True fit_path=True

positive=False

fitselfXyXyNone

Fit the model using X y as training data

Parameters

Xarraylike shape nsamples nfeatures Training data

yarraylike shape nsamples or nsamples ntargets Target values

Xyarraylike shape nsamples or nsamples ntargets optional Xy np.dot(XT y

1882 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

that can be precomputed It is useful only when the Gram matrix is precomputed

Returns

self object returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

622sklearnlinearmodel Generalized Linear Models 1883

scikitlearn user guide Release 0213

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

6226sklearnlinearmodel Lasso

classsklearnlinearmodel Lassoalpha10 fitinterceptTrue normalizeFalse precom

puteFalse copyXTrue maxiter1000 tol00001

warmstartFalse positiveFalse randomstateNone selec

tion'cyclic'

Linear Model trained with L1 prior as regularizer aka the Lasso

The optimization objective for Lasso is

$$\frac{1}{2} \|y - Xw\|_2^2 + \alpha \|w\|_1$$

Technically the Lasso model is optimizing the same objective function as the Elastic Net with l1ratio10

no L2 penalty

Read more in the User Guide

Parameters

alpha float optional Constant that multiplies the L1 term Defaults to 10 alpha 0 is

equivalent to an ordinary least square solved by the LinearRegression object For

numerical reasons using alpha 0 with theLasso object is not advised Given this

you should use the LinearRegression object

fitintercept boolean optional default True Whether to calculate the intercept for this model

If set to False no intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before

callingfit on an estimator with normalizeFalse

precompute True False arraylike defaultFalse Whether to use a precomputed Gram ma

trix to speed up calculations If set to auto let us decide The Gram matrix can also be

passed as argument For sparse input this option is always True to preserve sparsity

copyX boolean optional default True If True X will be copied else it may be overwritten

maxiter int optional The maximum number of iterations

tolfloat optional The tolerance for the optimization if the updates are smaller than tol the

optimization code checks the dual gap for optimality and continues until it is smaller than tol

warmstart bool optional When set to True reuse the solution of the previous call to fit as

initialization otherwise just erase the previous solution See the Glossary

positive bool optional When set to True forces the coefficients to be positive

1884 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

randomstate int RandomState instance or None optional default None The seed of the pseudo random number generator that selects a random feature to update If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom Used when selection 'random' selection str default 'cyclic' If set to 'random' a random coefficient is updated every iteration rather than looping over features sequentially by default This setting to 'random' often leads to significantly faster convergence especially when tol is higher than 1e4

Attributes

coef array shape nfeatures ntargets nfeatures parameter vector w in the cost function formula
sparsecoef scipy sparse matrix shape nfeatures 1 ntargets nfeatures sparse representation of the fitted coef
intercept float array shape ntargets independent term in decision function
niter int arraylike shape ntargets number of iterations run by the coordinate descent solver to reach the specified tolerance

See also

larspath
lassopath
LassoLars
LassoCV
LassoLarsCV
sklearn.decomposition.sparse_encode

Notes

The algorithm used to fit the model is coordinate descent
To avoid unnecessary memory duplication the X argument of the fit method should be directly passed as a Fortran contiguous numpy array

Examples

```
from sklearn import linear_model
clf = linear_model.Lasso(alpha=0.1)
clf.fit(X, y)
```

```
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=1000,
normalize=False, positive=False, precompute=False, random_state=None,
selection=cyclic, tol=0.0001, warm_start=False)
print(clf.coef_)
0.85 0
print(clf.intercept_)
0.15
622sklearn.linear_model.Generalized Linear Models 1885
```

Methods

fitself X y checkinput Fit model with coordinate descent
getparams self deep Get parameters for this estimator
path X y l1ratio eps nalphas Compute elastic net path with coordinate descent
predict self X Predict using the linear model
score self X y sampleweight Returns the coefficient of determination R2 of the pre
diction
setparams self params Set the parameters of this estimator
init selfalpha10 fitinterceptTrue normalizeFalse precomputeFalse copyXTrue
maxiter1000 tol000001 warmstartFalse positiveFalse randomstateNone se
lection'cyclic'
fitselfXycheckinputTrue
Fit model with coordinate descent

Parameters

Xndarray or scipysparse matrix nsamples nfeatures Data
yndarray shape nsamples or nsamples ntargets Target Will be cast to X's dtype
if necessary
checkinput boolean defaultTrue Allow to bypass several input checking Don't use
this parameter unless you know what you do

Notes

Coordinate descent is an algorithm that considers each column of data at a time hence it will automatically
convert the X input as a Fortrancontiguous numpy array if necessary
To avoid memory reallocation it is advised to allocate the initial data in memory directly using that format

getparams selfdeepTrue
Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values
staticpathXyl1ratio05 eps0001 nalphas100 alphasNone precompute'auto'
XyNone copyXTrue coefinitNone verboseFalse returnniterFalse posi
tiveFalse checkinputTrue params

Compute elastic net path with coordinate descent
The elastic net optimization function varies for mono and multioutputs
For monooutput tasks it is

1 2nsamples y Xw22
alpha l1ratio w1
05alpha1 l1ratio w22

scikitlearn user guide Release 0213

For multioutput tasks it is

1 2 nsamples Y XWFro2

alpha l1ratio W21

05alpha1 l1ratio WFro2

Where

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran

contiguous data to avoid unnecessary memory duplication If yis monooutput then X

can be sparse

yndarray shape nsamples or nsamples noutputs Target values

l1ratio float optional float between 0 and 1 passed to elastic net scaling between l1 and

l2 penalties l1ratio1 corresponds to the Lasso

eps float Length of the path eps1e3 means that alphamin alphamax

1e3

nalphas int optional Number of alphas along the regularization path

alphas ndarray optional List of alphas where to compute the models If None alphas are

set automatically

precompute True False ‘auto’ arraylike Whether to use a precomputed Gram matrix

to speed up calculations If set to auto let us decide The Gram matrix can also be

passed as argument

Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only

when the Gram matrix is precomputed

copyX boolean optional default True If True X will be copied else it may be over

written

coefinit array shape nfeatures None The initial values of the coefficients

verbose bool or integer Amount of verbosity

returnniter bool whether to return the number of iterations or not

positive bool default False If set to True forces coefficients to be positive Only allowed

whenyndim 1

checkinput bool default True Skip input validation checks including the Gram matrix

when provided assuming there are handled by the caller when checkinputFalse

params kwargs keyword arguments passed to the coordinate descent solver

Returns

alphas array shape nalphas The alphas along the path where models are computed

coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients

along the path

622sklearnlinearmodel Generalized Linear Models 1887

scikitlearn user guide Release 0213

dualgaps array shape nalphas The dual gaps at the end of the optimization for each alpha

niters arraylike shape nalphas The number of iterations taken by the coordinate descent optimizer to reach the specified tolerance for each alpha Is returned when returnniter is set to True

See also

MultiTaskElasticNet

MultiTaskElasticNetCV

ElasticNet

ElasticNetCV

Notes

For an example see exampleslinearmodelplotlassocoordinatedescentpathpy

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score is 10 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

1888 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses
multioutputuniformaverage
setparams selfparams
Set the parameters of this estimator
The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it's possible to update each component
of a nested object
Returns
self
sparsecoef
sparse representation of the fitted coef
Examples using sklearnlinearmodelLasso
•Compressive sensing tomography reconstruction with L1 prior Lasso
•Crossvalidation on diabetes Dataset Exercise
•Lasso on dense and sparse data
•Joint feature selection with multitask Lasso
•Lasso and Elastic Net for Sparse Signals
6227sklearnlinearmodel LassoLars
classsklearnlinearmodel LassoLars alpha10 fitinterceptTrue verboseFalse nor
malizeTrue precompute'auto' maxiter500
eps2220446049250313e16 copyXTrue
fitpathTrue positiveFalse
Lasso model fit with Least Angle Regression aka Lars
It is a Linear Model trained with an L1 prior as regularizer
The optimization objective for Lasso is
$$\frac{1}{2} \|y - Xw\|_2^2 + \alpha \|w\|_1$$

Read more in the User Guide
Parameters
alpha float Constant that multiplies the penalty term Defaults to 10 alpha 0 is equiv
alent to an ordinary least square solved by LinearRegression For numerical reasons
usingalpha 0 with the LassoLars object is not advised and you should prefer the Lin
earRegression object
fitintercept boolean whether to calculate the intercept for this model If set to false no
intercept will be used in calculations eg data is expected to be already centered
verbose boolean or integer optional Sets the verbosity amount
normalize boolean optional default True This parameter is ignored when fitintercept
is set to False If True the regressors X will be normalized before regression by sub
tracting the mean and dividing by the l2norm If you wish to standardize please use
622sklearnlinearmodel Generalized Linear Models 1889

scikitlearn user guide Release 0213

sklearn.preprocessing.StandardScaler before calling fit on an estimator
with_normalize=False

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to speed up calculations If set to auto let us decide The Gram matrix can also be passed as argument

max_iter integer optional Maximum number of iterations to perform

eps float optional The machineprecision regularization in the computation of the Cholesky diagonal factors Increase this for very illconditioned systems Unlike the tol parameter in some iterative optimizationbased algorithms this parameter does not control the tolerance of the optimization

copy_X boolean optional default True If True X will be copied else it may be overwritten

fit_path boolean If True the full path is stored in the coef_path attribute If you compute the solution for a large problem or many targets setting fit_path to False will lead to a speedup especially with a small alpha

positive boolean default False Restrict coefficients to be ≥ 0 Be aware that you might want to remove fit_intercept which is set True by default Under the positive restriction the model coefficients will not converge to the ordinaryleast squares solution for small values of alpha Only coefficients up to the smallest alpha value α_{min} when fit_path=True reached by the stepwise LarsLasso algorithm are typically in congruence with the solution of the coordinate descent Lasso estimator

Attributes

alphas array shape n_targets list of n_targets such arrays Maximum of covariances in absolute value at each iteration n_alphas is either max_iter n_features or the number of nodes in the path with correlation greater than alpha whichever is smaller
active_list_length n_alphas list of n_targets such lists Indices of active variables at the end of the path

coef_path array shape n_features n_alphas 1 or list If a list is passed it's expected to be one of n_targets such arrays The varying values of the coefficients along the path It is not present if the fit_path parameter is False

coef array shape n_features or n_targets n_features Parameter vector w in the formula
tion formula

intercept float array shape n_targets Independent term in decision function

n_iter arraylike or int The number of iterations taken by lars_path to find the grid of alphas for each target

See also

lars_path

lasso_path

Lasso

LassoCV

LassoLarsCV

LassoLarsIC

sklearn.decomposition.sparse_encode

1890 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

```
from sklearn import linearmodel
reg = linearmodel.LassoLarsalpha(0.01)
reg.fit(1 1 0 0 1 1 1 0 1)
```

```
LassoLarsalpha(0.01, copy_X=True, eps=1e-05, fit_intercept=True,
fit_path=True, max_iter=500, normalize=True, positive=False,
precompute='auto', verbose=False)
print(reg.coef_)
0.0963257
```

Methods

```
fit(self, X, y, Xy=None) Fit the model using X, y as training data
get_params(self, deep=True) Get parameters for this estimator
predict(self, X) Predict using the linear model
score(self, X, y, sample_weight=None) Returns the coefficient of determination R^2 of the prediction
set_params(self, **params) Set the parameters of this estimator
init(self, alpha=1.0, fit_intercept=True, verbose=False, normalize=True, precompute='auto',
max_iter=500, eps=2.220446049250313e-16, copy_X=True, fit_path=True, positive=False)
fit(self, X, y, Xy=None)
```

Fit the model using X, y as training data

Parameters

```
X: arraylike shape (n_samples, n_features) Training data
y: arraylike shape (n_samples, or (n_samples, n_targets)) Target values
Xy: arraylike shape (n_samples, or (n_samples, n_targets)) optional Xy = np.dot(X, y)
that can be precomputed. It is useful only when the Gram matrix is precomputed
```

Returns

```
self: object returns an instance of self
```

```
get_params(self, deep=True)
```

Get parameters for this estimator

Parameters

```
deep: boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
```

Returns

```
params: mapping of string to any Parameter names mapped to their values
```

```
predict(self, X)
```

Predict using the linear model

Parameters

```
X: arraylike or sparse matrix shape (n_samples, n_features) Samples
```

scikitlearn user guide Release 0213

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

6228sklearnlinearmodel LinearRegression

classsklearnlinearmodel LinearRegression fitinterceptTrue normalizeFalse

copyXTrue njobsNone

Ordinary least squares Linear Regression

Parameters

fitintercept boolean optional default True whether to calculate the intercept for this model

If set to False no intercept will be used in calculations eg data is expected to be already

centered

1892 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

normalize boolean optional default False This parameter is ignored when
fitintercept is set to False If True the regressors X will be normalized be
fore regression by subtracting the mean and dividing by the l2norm If you wish to
standardize please use sklearnpreprocessingStandardScaler before
callingfit on an estimator with normalizeFalse
copyX boolean optional default True If True X will be copied else it may be overwritten
njobs int or None optional defaultNone The number of jobs to use for the computation
This will only provide speedup for ntargets 1 and sufficient large problems None means
1 unless in a joblibparallelbackend context1means using all processors See
Glossary for more details

Attributes
coef array shape nfeatures or ntargets nfeatures Estimated coefficients for the linear
regression problem If multiple targets are passed during the fit y 2D this is a 2D array of
shape ntargets nfeatures while if only one target is passed this is a 1D array of length
nfeatures

intercept array Independent term in the linear model

Notes
From the implementation point of view this is just plain Ordinary Least Squares scipy.linalg.lstsq wrapped as
a predictor object

Examples
import numpy as np
from sklearnlinearmodel import LinearRegression
X nparray1 1 1 2 2 2 2 3
y 1 x0 2 x1 3
y npdotX nparray1 2 3
reg LinearRegressionfitX y
regscoreX y
10
regcoef
array1 2
regintercept
30000
regpredictnparray3 5
array16

Methods
fitself X y sampleweight Fit linear model
getparams self deep Get parameters for this estimator
predict self X Predict using the linear model
score self X y sampleweight Returns the coefficient of determination R2 of the pre
diction
setparams self params Set the parameters of this estimator
622sklearnlinearmodel Generalized Linear Models 1893

scikitlearn user guide Release 0213

init selffitinterceptTrue normalizeFalse copyXTrue njobsNone

fitselfXysampleweightNone

Fit linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Training data

yarraylike shape nsamples ntargets Target values Will be cast to X's dtype if nec

essary

sampleweight numpy array of shape nsamples Individual weights for each sample

New in version 017 parameter sampleweight support to LinearRegression

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

1894 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnlinearmodelLinearRegression

- Isotonic Regression
 - Face completion with a multioutput estimators
 - Plot individual and voting regression predictions
 - Ordinary Least Squares and Ridge Regression Variance
 - Logistic function
 - Linear Regression Example
 - Robust linear model estimation using RANSAC
 - Sparsity Example Fitting only features 1 and 2
 - TheilSen Regression
 - Robust linear estimator fitting
 - Automatic Relevance Determination Regression ARD
 - Bayesian Ridge Regression
 - Plotting CrossValidated Predictions
 - Underfitting vs Overfitting
 - Using KBinsDiscretizer to discretize continuous features
- 622sklearnlinearmodel Generalized Linear Models 1895

scikitlearn user guide Release 0213

6229sklearnlinearmodel LogisticRegression

classsklearnlinearmodel LogisticRegression penalty'12' dualFalse tol00001

C10 fitinterceptTrue inter

ceptscaling1 classweightNone

randomstateNone solver'warn'

maxiter100 multiclass'warn' ver

bose0 warmstartFalse njobsNone

l1ratioNone

Logistic Regression aka logit MaxEnt classifier

In the multiclass case the training algorithm uses the onevsrest OvR scheme if the 'multiclass' option is

set to 'ovr' and uses the crossentropy loss if the 'multiclass' option is set to 'multinomial' Currently the

'multinomial' option is supported only by the 'lbfgs' 'sag' 'saga' and 'newtoncg' solvers

This class implements regularized logistic regression using the 'liblinear' library 'newtoncg' 'sag' 'saga' and

'lbfgs' solvers Note that regularization is applied by default It can handle both dense and sparse input Use

Cordered arrays or CSR (Compressed Sparse Row) matrices containing 64bit floats for optimal performance any other input format will

be converted and copied

The 'newtoncg' 'sag' and 'lbfgs' solvers support only L2 regularization with primal formulation or no reg

ularization The 'liblinear' solver supports both L1 and L2 regularization with a dual formulation only for the

L2 penalty The ElasticNet regularization is only supported by the 'saga' solver

Read more in the User Guide

Parameters

penalty str 'l1' 'l2' 'elasticnet' or 'none' optional default'l2' Used to specify the norm

used in the penalization The 'newtoncg' 'sag' and 'lbfgs' solvers support only l2 penal

ties 'elasticnet' is only supported by the 'saga' solver If 'none' not supported by the

liblinear solver no regularization is applied

New in version 019 l1 penalty with SAGA solver allowing 'multinomial' L1

dual bool optional defaultFalse Dual or primal formulation Dual formulation is only

implemented for l2 penalty with liblinear solver Prefer dualFalse when nsamples

nfeatures

tolfloat optional default1e4 Tolerance for stopping criteria

Cfloat optional default10 Inverse of regularization strength must be a positive float Like

in support vector machines smaller values specify stronger regularization

fitintercept bool optional defaultTrue Specifies if a constant aka bias or intercept

should be added to the decision function

interceptscaling float optional default1 Useful only when the solver 'liblinear' is used

and selffitintercept is set to True In this case x becomes x selfinterceptscaling

ie a "synthetic" feature with constant value equal to interceptscaling is ap

pended to the instance vector The intercept becomes interceptscaling

syntheticfeatureweight

Note the synthetic feature weight is subject to l1l2 regularization as all other features To

lessen the effect of regularization on synthetic feature weight and therefore on the intercept

interceptscaling has to be increased

classweight dict or 'balanced' optional defaultNone Weights associated with classes in

the formclasslabel weight If not given all classes are supposed to have

weight one

1896 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as $n_{\text{samples}} / n_{\text{classes}}$ n_{p} $n_{\text{bincounty}}$

Note that these weights will be multiplied with sampleweight passed through the fit method if sampleweight is specified

New in version 017 classweight ‘balanced’

randomstate int RandomState instance or None optional default None The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom Used when solver ‘sag’ or ‘liblinear’

solver str ‘newtoncg’ ‘lbfgs’ ‘liblinear’ ‘sag’ ‘saga’ optional default ‘liblinear’ Algorithm to use in the optimization problem

- For small datasets ‘liblinear’ is a good choice whereas ‘sag’ and ‘saga’ are faster for large ones

- For multiclass problems only ‘newtoncg’ ‘sag’ ‘saga’ and ‘lbfgs’ handle multinomial loss ‘liblinear’ is limited to one versus rest schemes

- ‘newtoncg’ ‘lbfgs’ ‘sag’ and ‘saga’ handle L2 or no penalty

- ‘liblinear’ and ‘saga’ also handle L1 penalty

- ‘saga’ also supports ‘elasticnet’ penalty

- ‘liblinear’ does not handle no penalty

Note that ‘sag’ and ‘saga’ fast convergence is only guaranteed on features with approximately the same scale You can preprocess the data with a scaler from sklearn.preprocessing

New in version 017 Stochastic Average Gradient descent solver

New in version 019 SAGA solver

Changed in version 020 Default will change from ‘liblinear’ to ‘lbfgs’ in 022

maxiter int optional default 100 Maximum number of iterations taken for the solvers to converge

multiclass str ‘ovr’ ‘multinomial’ ‘auto’ optional default ‘ovr’ If the option chosen is ‘ovr’ then a binary problem is fit for each label For ‘multinomial’ the loss minimised is the multinomial loss fit across the entire probability distribution even when the data is binary ‘multinomial’ is unavailable when solver ‘liblinear’ ‘auto’ selects ‘ovr’ if the data is binary or if solver ‘liblinear’ and otherwise selects ‘multinomial’

New in version 018 Stochastic Average Gradient descent solver for ‘multinomial’ case

Changed in version 020 Default will change from ‘ovr’ to ‘auto’ in 022

verbose int optional default 0 For the liblinear and lbfgs solvers set verbose to any positive number for verbosity

warmstart bool optional default False When set to True reuse the solution of the previous call to fit as initialization otherwise just erase the previous solution Useless for liblinear solver See the Glossary

New in version 017 warmstart to support lbfgs newtoncg sagsaga solvers

njobs int or None optional default None Number of CPU cores used when parallelizing over classes if multiclass ‘ovr’ This parameter is ignored when the solver is set to

622sklearnlinear model Generalized Linear Models 1897

scikitlearn user guide Release 0213

‘liblinear’ regardless of whether ‘multiclass’ is specified or not None means 1 unless in
ajoblibparallelbackend context1means using all processors See Glossary
for more details

l1ratio float or None optional defaultNone The ElasticNet mixing parameter with
0 l1ratio 1 Only used if penaltyelasticnet Setting
l1ratio0 is equivalent to using penaltyl2 while setting l1ratio1 is
equivalent to using penaltyl1 For0 l1ratio 1 the penalty is a combi
nation of L1 and L2

Attributes

classes array shape nclasses A list of class labels known to the classifier
coef array shape 1 nfeatures or nclasses nfeatures Coefficient of the features in the
decision function

coef is of shape 1 nfeatures when the given problem is binary In particular when
multiclassmultinomial coef corresponds to outcome 1 True and coef
corresponds to outcome 0 False

intercept array shape 1 or nclasses Intercept aka bias added to the decision func
tion

l1fitintercept is set to False the intercept is set to zero intercept
is of shape 1 when the given problem is binary In particular when
multiclassmultinomial intercept corresponds to outcome 1 True and
intercept corresponds to outcome 0 False

niter array shape nclasses or 1 Actual number of iterations for all classes If binary
or multinomial it returns only 1 element For liblinear solver only the maximum number
of iteration across all classes is given

Changed in version 020 In SciPy 100 the number of lbfgs iterations may exceed
maxiter niter will now report at most maxiter

See also
SGDClassifier incrementally trained logistic regression when given the parameter losslog
LogisticRegressionCV Logistic regression with builtin cross validation

Notes
The underlying C implementation uses a random number generator to select features when fitting the model
It is thus not uncommon to have slightly different results for the same input data If that happens try with a
smaller tol parameter

Predict output may not match that of standalone liblinear in certain cases See differences from liblinear in the
narrative documentation

References
LIBLINEAR – A Library for Large Linear Classification <https://www.cs.tufts.edu/ncv/liblinear/>
SAG – Mark Schmidt Nicolas Le Roux and Francis Bach Minimizing Finite Sums with the Stochastic Av
erage Gradient <http://hal.inria.fr/hal-00860051/document>
1898 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

SAGA – Defazio A Bach F LacosteJulien S 2014 SAGA A Fast Incremental Gradient Method With Support for NonStrongly Convex Composite Objectives <https://arxiv.org/abs/1407.0202>

HsiangFu Yu FangLan Huang Chihjen Lin 2011 Dual coordinate descent methods for logistic regression and maximum entropy models Machine Learning 85:124175 <https://www.cs.cmu.edu/papers/08/ml08-dual.pdf>

Examples

```
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
X, y = load_iris(return_Xy=True)
clf = LogisticRegression(random_state=0, solver='lbfgs')
clf.fit(X, y)
clf.predict(X)
array([0, 0, 0])
clf.predict_proba(X)
array([[0.98, 0.01, 0.01],
       [0.18, 0.82, 0.0],
       [0.97, 0.28, 0.0]])
clf.score(X, y)
0.97
```

Methods

decision_function(self, X) Predict confidence scores for samples

densify(self) Convert coefficient matrix to dense array format

fit(self, X, y, sample_weight) Fit the model according to the given training data

get_params(self, deep=True) Get parameters for this estimator

predict(self, X) Predict class labels for samples in X

predict_log_proba(self, X) Log of probability estimates

predict_proba(self, X) Probability estimates

score(self, X, y, sample_weight) Returns the mean accuracy on the given test data and labels

set_params(self, **params) Set the parameters of this estimator

sparsify(self) Convert coefficient matrix to sparse format

init(self, penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='warn', max_iter=100, multi_class='warn', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None)

decision_function(self, X)

Predict confidence scores for samples

The confidence score for a sample is the signed distance of that sample to the hyperplane

Parameters

X: array-like or sparse matrix shape (n_samples, n_features) Samples

Returns

array shape (n_samples, n_classes - 1) Confidence scores per sample class combination In the binary case confidence score for self.classes_[1] where 0 means this class would be predicted

622sklearn.linear_model Generalized Linear Models 1899

scikitlearn user guide Release 0213

densifyself

Convert coefficient matrix to dense array format

Converts the coef member back to a numpyndarray This is the default format of coef and is required for fitting so calling this method is only required on models that have previously been sparsified otherwise it is a noop

Returns

self estimator

fitselfXysampleweightNone

Fit the model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target vector relative to X

sampleweight arraylike shape nsamples optional Array of weights that are assigned to individual samples If not provided then each sample is given unit weight

New in version 017 sampleweight support to LogisticRegression

Returns

self object

Notes

The SAGA solver supports both float64 and float32 bit arrays

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Predicted class label per sample

predictlogproba selfX

Log of probability estimates

The returned estimates for all classes are ordered by the label of classes

Parameters

Xarraylike shape nsamples nfeatures

1900 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

Tarraylike shape nsamples nclasses Returns the logprobability of the sample for each class in the model where classes are ordered as they are in selfclasses

predictproba selfX

Probability estimates

The returned estimates for all classes are ordered by the label of classes

For a multiclass problem if multiclass is set to be “multinomial” the softmax function is used to find the predicted probability of each class Else use a onevsrest approach ie calculate the probability of each class assuming it to be positive using the logistic function and normalize these values across all the classes

Parameters

Xarraylike shape nsamples nfeatures

Returns

Tarraylike shape nsamples nclasses Returns the probability of the sample for each class in the model where classes are ordered as they are in selfclasses

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

sparsify self

Convert coefficient matrix to sparse format

Converts the coef member to a scipysparse matrix which for L1regularized models can be much more memory and storageefficient than the usual numpyndarray representation

Theintercept member is not converted

Returns

self estimator

622sklearnlinearmodel Generalized Linear Models 1901

Notes

For nonsparse models ie when there are not many zeros in coef this may actually increase memory usage so use this method with care A rule of thumb is that the number of zero elements which can be computed with coef_0sum must be more than 50 for this to provide significant benefits After calling this method further fitting with the partialfit method if any will not work until you call densify

Examples using sklearnlinearmodelLogisticRegression

- Compact estimator representations
- Comparison of Calibration of Classifiers
- Probability Calibration curves
- Plot classification probability
- Column Transformer with Mixed Types
- Plot class probabilities calculated by the VotingClassifier
- Feature transformations with ensembles of trees
- Digits Classification Exercise
- Regularization path of L1 Logistic Regression
- Logistic function
- Logistic Regression 3class Classifier
- Comparing various online solvers
- MNIST classification using multinomial logistic L1
- Plot multinomial and OnevsRest Logistic Regression
- L1 Penalty and Sparsity in Logistic Regression
- Multiclass sparse logisitic regression on newgroups20
- Classifier Chain
- Restricted Boltzmann Machine features for digit classification
- Feature discretization

62210sklearnlinearmodel MultiTaskLasso
classsklearnlinearmodel MultiTaskLasso alpha10 fitinterceptTrue normalizeFalse
copyXTrue maxiter1000 tol00001
warmstartFalse randomstateNone selec
tion'cyclic'

Multitask Lasso model trained with L1L2 mixednorm as regularizer

The optimization objective for Lasso is

$$\frac{1}{2} \|Y - XW\|_2^2 + \alpha \|W\|_1$$

Where

scikitlearn user guide Release 0213

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

Parameters

alpha float optional Constant that multiplies the L1L2 term Defaults to 10

fitintercept boolean whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before

callingfit on an estimator with normalizeFalse

copyX boolean optional default True If True X will be copied else it may be overwritten

maxiter int optional The maximum number of iterations

tolfloat optional The tolerance for the optimization if the updates are smaller than tol the

optimization code checks the dual gap for optimality and continues until it is smaller than

tol

warmstart bool optional When set to True reuse the solution of the previous call to fit as

initialization otherwise just erase the previous solution See the Glossary

randomstate int RandomState instance or None optional default None The seed of the

pseudo random number generator that selects a random feature to update If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom Used when selection 'random'

selection str default 'cyclic' If set to 'random' a random coefficient is updated every iteration

rather than looping over features sequentially by default This setting to 'random' often

leads to significantly faster convergence especially when tol is higher than 1e4

Attributes

coef array shape ntasks nfeatures Parameter vector W in the cost function formula

Note thatcoef stores the transpose of WW^T

intercept array shape ntasks independent term in decision function

niter int number of iterations run by the coordinate descent solver to reach the specified

tolerance

See also

MultiTaskLasso Multitask L1L2 Lasso with builtin crossvalidation

Lasso

MultiTaskElasticNet

Notes

The algorithm used to fit the model is coordinate descent

622sklearnlinearmodel Generalized Linear Models 1903

scikitlearn user guide Release 0213

To avoid unnecessary memory duplication the X argument of the fit method should be directly passed as a Fortrancontiguous numpy array

Examples

```
from sklearn import linearmodel
clf = linearmodelMultiTaskLassoalpha01
clf.fit00 1 1 2 2 0 0 1 1 2 2
```

MultiTaskLassoalpha01 copyXTrue fitinterceptTrue maxiter1000
normalizeFalse randomstateNone selectioncyclic tol00001
warmstartFalse

```
printclfcoef
089393398 0
089393398 0
printclfintercept
010606602 010606602
```

Methods

fitself X y Fit MultiTaskElasticNet model with coordinate descent
getparams self deep Get parameters for this estimator
path X y l1ratio eps nalphas Compute elastic net path with coordinate descent
predict self X Predict using the linear model
score self X y sampleweight Returns the coefficient of determination R2 of the pre
diction
setparams self params Set the parameters of this estimator
init selfalpha10 fitinterceptTrue normalizeFalse copyXTrue maxiter1000
tol00001 warmstartFalse randomstateNone selection'cyclic'

fitselfXy
Fit MultiTaskElasticNet model with coordinate descent

Parameters

Xndarray shape nsamples nfeatures Data
yndarray shape nsamples ntasks Target Will be cast to X's dtype if necessary

Notes

Coordinate descent is an algorithm that considers each column of data at a time hence it will automatically
convert the X input as a Fortrancontiguous numpy array if necessary

To avoid memory reallocation it is advised to allocate the initial data in memory directly using that format

getparams selfdeepTrue
Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

scikitlearn user guide Release 0213

Returns

params mapping of string to any Parameter names mapped to their values
pathXyl1ratio05 eps0001 nalphas100 alphasNone precompute'auto' XyNone
copyXTrue coefinitNone verboseFalse returnniterFalse positiveFalse
checkinputTrue params

Compute elastic net path with coordinate descent

The elastic net optimization function varies for mono and multioutputs

For monooutput tasks it is

1 2nsamples y Xw22

alpha l1ratio w1

05alpha1 l1ratio w22

For multioutput tasks it is

1 2 nsamples Y XWFro2

alpha l1ratio W21

05alpha1 l1ratio WFro2

Where

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran

contiguous data to avoid unnecessary memory duplication If yis monooutput then X

can be sparse

yndarray shape nsamples or nsamples noutputs Target values

l1ratio float optional float between 0 and 1 passed to elastic net scaling between l1 and

l2 penalties l1ratio1 corresponds to the Lasso

eps float Length of the path eps1e3 means that alphamin alphamax

1e3

nalphas int optional Number of alphas along the regularization path

alphas ndarray optional List of alphas where to compute the models If None alphas are
set automatically

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix
to speed up calculations If set to auto let us decide The Gram matrix can also be

passed as argument

Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only
when the Gram matrix is precomputed

copyX boolean optional default True If True X will be copied else it may be over
written

coefinit array shape nfeatures None The initial values of the coefficients

verbose bool or integer Amount of verbosity

622sklearnlinearmodel Generalized Linear Models 1905

scikitlearn user guide Release 0213

returnniter bool whether to return the number of iterations or not
positive bool default False If set to True forces coefficients to be positive Only allowed
whenyndim 1
checkinput bool default True Skip input validation checks including the Gram matrix
when provided assuming there are handled by the caller when checkinputFalse
params kwargs keyword arguments passed to the coordinate descent solver
Returns

alphas array shape nalphas The alphas along the path where models are computed
coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients
along the path
dualgaps array shape nalphas The dual gaps at the end of the optimization for each
alpha
niters arraylike shape nalphas The number of iterations taken by the coordinate
descent optimizer to reach the specified tolerance for each alpha Is returned when
returnniter is set to True

See also
MultiTaskElasticNet
MultiTaskElasticNetCV
ElasticNet
ElasticNetCV

Notes
For an example see exampleslinearmodelplotlassocoordinatedescentpathpy
predictselfX

Predict using the linear model

Parameters
Xarraylike or sparse matrix shape nsamples nfeatures Samples
Returns

Carray shape nsamples Returns predicted values
scoreselfXysampleweightNone
Returns the coefficient of determination R2 of the prediction
The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$
 $2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score
is 10 and it can be negative because the model can be arbitrarily worse A constant model that always
predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters
Xarraylike shape nsamples nfeatures Test samples For some estimators this may
be a precomputed kernel matrix instead shape nsamples nsamplesfitted where
nsamplesfitted is the number of samples used in the fitting for the estimator
yarraylike shape nsamples or nsamples noutputs True values for X
1906 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

sparsecoef

sparse representation of the fitted coef

Examples using sklearnlinearmodelMultiTaskLasso

•Joint feature selection with multitask Lasso

62211sklearnlinearmodel MultiTaskElasticNet

classssklearnlinearmodel MultiTaskElasticNet alpha10 l1ratio05

fitinterceptTrue normalizeFalse

copyXTrue maxiter1000 tol00001

warmstartFalse randomstateNone

selection'cyclic'

Multitask ElasticNet model trained with L1L2 mixednorm as regularizer

The optimization objective for MultiTaskElasticNet is

$\frac{1}{2} \|Y - XW\|_F^2$

$+\frac{\alpha}{2} \|W\|_1$

Where

$\|W\|_1 = \sum_i \sqrt{\sum_j w_{ij}^2}$
ie the sum of norm of each row

Read more in the User Guide

Parameters

622sklearnlinearmodel Generalized Linear Models 1907

scikitlearn user guide Release 0213

alpha float optional Constant that multiplies the L1L2 term Defaults to 10

l1ratio float The ElasticNet mixing parameter with 0 l1ratio 1 For l1ratio 1 the

penalty is an L1L2 penalty For l1ratio 0 it is an L2 penalty For 0 l1ratio

1 the penalty is a combination of L1L2 and L2

fitintercept boolean whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before

callingfit on an estimator with normalizeFalse

copyX boolean optional default True If True X will be copied else it may be overwritten

maxiter int optional The maximum number of iterations

tolfloat optional The tolerance for the optimization if the updates are smaller than tol the

optimization code checks the dual gap for optimality and continues until it is smaller than

tol

warmstart bool optional When set to True reuse the solution of the previous call to fit as

initialization otherwise just erase the previous solution See the Glossary

randomstate int RandomState instance or None optional default None The seed of the

pseudo random number generator that selects a random feature to update If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom Used when selection 'random'

selection str default 'cyclic' If set to 'random' a random coefficient is updated every iteration

rather than looping over features sequentially by default This setting to 'random' often

leads to significantly faster convergence especially when tol is higher than 1e4

Attributes

intercept array shape ntasks Independent term in decision function

coef array shape ntasks nfeatures Parameter vector W in the cost function formula If

a 1D y is passed in at fit non multitask usage coef is then a 1D array Note that coef

stores the transpose of WWT

niter int number of iterations run by the coordinate descent solver to reach the specified

tolerance

See also

MultiTaskElasticNet Multitask L1L2 ElasticNet with builtin crossvalidation

ElasticNet

MultiTaskLasso

Notes

The algorithm used to fit the model is coordinate descent

To avoid unnecessary memory duplication the X argument of the fit method should be directly passed as a

Fortrancontiguous numpy array

1908 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

from sklearn import linearmodel

clf = linearmodel.MultiTaskElasticNet(alpha=0.1

l1_ratio=0.5) # 1 1 2 2 0 0 1 1 2 2

MultiTaskElasticNet(alpha=0.1, copy_X=True, fit_intercept=True,
l1_ratio=0.5, max_iter=1000, normalize=False, random_state=None,
selection='cyclic', tol=0.0001, warm_start=False)

print(clf.coef_)

0.45663524 0.45612256

0.45663524 0.45612256

print(clf.intercept_)

0.0872422 0.0872422

Methods

fit(self, X, y) Fit MultiTaskElasticNet model with coordinate descent

get_params(self, deep=True) Get parameters for this estimator

path(self, X, y, l1_ratio, eps, n_alphas) Compute elastic net path with coordinate descent

predict(self, X) Predict using the linear model

score(self, X, y, sample_weight) Returns the coefficient of determination R² of the pre-

diction

set_params(self, **params) Set the parameters of this estimator

init(self, alpha=0.1, l1_ratio=0.5, fit_intercept=True, normalize=False, copy_X=True,

max_iter=1000, tol=0.0001, warm_start=False, random_state=None, selection='cyclic')

fit(self, X, y)

Fit MultiTaskElasticNet model with coordinate descent

Parameters

X ndarray shape (n_samples, n_features) Data

y ndarray shape (n_samples, n_tasks) Target Will be cast to X's dtype if necessary

Notes

Coordinate descent is an algorithm that considers each column of data at a time hence it will automatically

convert the X input as a Fortran contiguous numpy array if necessary

To avoid memory reallocation it is advised to allocate the initial data in memory directly using that format

get_params(self, deep=True)

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

622sklearn.linear_model.Generalized Linear Models 1909

scikitlearn user guide Release 0213

pathXyl1ratio05 eps0001 nalphas100 alphasNone precompute'auto' XyNone
copyXTrue coefinitNone verboseFalse returnniterFalse positiveFalse
checkinputTrue params

Compute elastic net path with coordinate descent

The elastic net optimization function varies for mono and multioutputs

For monooutput tasks it is

1 2nsamples y Xw22
alpha l1ratio w1
05alpha1 l1ratio w22

For multioutput tasks it is

1 2 nsamples Y XWFro2
alpha l1ratio W21
05alpha1 l1ratio WFro2

Where

W21 sumi sqrtsumj wij2
ie the sum of norm of each row

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran
contiguous data to avoid unnecessary memory duplication If yis monooutput then X
can be sparse

yndarray shape nsamples or nsamples noutputs Target values

l1ratio float optional float between 0 and 1 passed to elastic net scaling between l1 and
l2 penalties l1ratio1 corresponds to the Lasso

eps float Length of the path eps1e3 means that alphamin alphamax
1e3

nalphas int optional Number of alphas along the regularization path

alphas ndarray optional List of alphas where to compute the models If None alphas are
set automatically

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix
to speed up calculations If set to auto let us decide The Gram matrix can also be
passed as argument

Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only
when the Gram matrix is precomputed

copyX boolean optional default True If True X will be copied else it may be over
written

coefinit array shape nfeatures None The initial values of the coefficients

verbose bool or integer Amount of verbosity

returnniter bool whether to return the number of iterations or not

positive bool default False If set to True forces coefficients to be positive Only allowed
whenyndim 1

1910 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

checkinput bool default True Skip input validation checks including the Gram matrix when provided assuming there are handled by the caller when checkinputFalse

params kwargs keyword arguments passed to the coordinate descent solver

Returns

alphas array shape nalphas The alphas along the path where models are computed

coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients along the path

dualgaps array shape nalphas The dual gaps at the end of the optimization for each alpha

niters arraylike shape nalphas The number of iterations taken by the coordinate descent optimizer to reach the specified tolerance for each alpha Is returned when returnniter is set to True

See also

MultiTaskElasticNet

MultiTaskElasticNetCV

ElasticNet

ElasticNetCV

Notes

For an example see exampleslinearmodelplotlassocoordinatedescentpathpy

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

622sklearnlinearmodel Generalized Linear Models 1911

scikitlearn user guide Release 0213

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage from version 023 to keep consistent with metricsr2score This will influence the score method of all the multioutput regressors except for multioutputMultiOutputRegressor To specify the default value manually and avoid the warning please either call metricsr2score directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

sparsecoef

sparse representation of the fitted coef

62212sklearnlinearmodel OrthogonalMatchingPursuit

classssklearnlinearmodel OrthogonalMatchingPursuit nnonzerocoefsNone

tolNone fitinterceptTrue nor

malizeTrue precompute'auto'

Orthogonal Matching Pursuit model OMP

Read more in the User Guide

Parameters

nnonzerocoefs int optional Desired number of nonzero entries in the solution If None

by default this value is set to 10 of nfeatures

tolfloat optional Maximum norm of the residual If not None overrides nnonzerocoefs

fitintercept boolean optional whether to calculate the intercept for this model If set to false

no intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default True This parameter is ignored when fitintercept

is set to False If True the regressors X will be normalized before regression by sub

tracting the mean and dividing by the l2norm If you wish to standardize please use

sklearnpreprocessingStandardScaler before calling fit on an estimator

withnormalizeFalse

precompute True False 'auto' default 'auto' Whether to use a precomputed Gram and

Xy matrix to speed up calculations Improves performance when ntargets ornsamples is

very large Note that if you already have such matrices you can pass them directly to the fit

method

Attributes

coef array shape nfeatures or ntargets nfeatures parameter vector w in the formula

intercept float or array shape ntargets independent term in decision function

1912 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
niter int or arraylike Number of active features across every target
See also
orthogonalmp
orthogonalmpgram
larspath
Lars
LassoLars
decompositionsparseencode
OrthogonalMatchingPursuitCV
Notes
Orthogonal matching pursuit was introduced in G Mallat Z Zhang Matching pursuits with timefrequency dictionaries IEEE Transactions on Signal Processing V ol 41 No 12 December 1993 pp 33973415
<http://blanchepolytechnique.fr/mallat/papiers/MallatPursuit93.pdf>
This implementation is based on Rubinstein R Zibulevsky M and Elad M Efficient Implementation of the KSVD Algorithm using Batch Orthogonal Matching Pursuit Technical Report CS Technion April 2008
<https://www.cs.technion.ac.il/~ronrubin/Publications/KSVDOMPv2.pdf>
Examples
from sklearn.linear_model import OrthogonalMatchingPursuit
from sklearn.datasets import make_regression
X, y = make_regression(n_samples=1000, n_features=10, noise=0.1, random_state=0)
reg = OrthogonalMatchingPursuit()
reg.fit(X, y)
reg.score(X, y)
0.9991
reg.predict(X)
array([78.3854])
Methods
fit(self, X, y) Fit the model using X, y as training data
get_params(self, deep=True) Get parameters for this estimator
predict(self, X) Predict using the linear model
score(self, X, y, sample_weight) Returns the coefficient of determination R² of the prediction
set_params(self, **params) Set the parameters of this estimator
init(self, n_nonzero_coefs=None, tol=None, fit_intercept=True, normalize=True, precompute=None, verbose=0)
fit(self, X, y) Fit the model using X, y as training data
Parameters
622sklearn.linear_model Generalized Linear Models 1913

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures Training data

yarraylike shape nsamples or nsamples ntargets Target values Will be cast to X's dtype if necessary

Returns

self object returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 0.23 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

1914 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnlinearmodelOrthogonalMatchingPursuit

•Orthogonal Matching Pursuit

62213sklearnlinearmodel PassiveAggressiveClassifier

classssklearnlinearmodel PassiveAggressiveClassifier C10 fitinterceptTrue

maxiter1000 tol0001

earlystoppingFalse

validationfraction01

niternochange5

shuffleTrue verbose0

loss'hinge' njobsNone

randomstateNone

warmstartFalse

classweightNone aver

ageFalse

Passive Aggressive Classifier

Read more in the User Guide

Parameters

Cfloat Maximum step size regularization Defaults to 10

fitintercept bool defaultFalse Whether the intercept should be estimated or not If False the data is assumed to be already centered

maxiter int optional default1000 The maximum number of passes over the training data

aka epochs It only impacts the behavior in the fit method and not the partialfit

New in version 019

tolfloat or None optional default1e3 The stopping criterion If it is not None the iterations will stop when loss previousloss tol

New in version 019

earlystopping bool defaultFalse Whether to use early stopping to terminate training when validation score is not improving If set to True it will automatically set aside a stratified fraction of training data as validation and terminate training when validation score is not improving by at least tol for niternochange consecutive epochs

New in version 020

validationfraction float default01 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if earlystopping is True

New in version 020

622sklearnlinearmodel Generalized Linear Models 1915

scikitlearn user guide Release 0213

niternochange int default5 Number of iterations with no improvement to wait before early stopping

New in version 020

shuffle bool defaultTrue Whether or not the training data should be shuffled after each epoch

verbose integer optional The verbosity level

loss string optional The loss function to be used hinge equivalent to PAI in the reference paper squaredhinge equivalent to PAII in the reference paper

njobs int or None optional defaultNone The number of CPUs to use to do the OV A One Versus All for multiclass problems computation None means 1 unless in a joblib parallelbackend context1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

warmstart bool optional When set to True reuse the solution of the previous call to fit as initialization otherwise just erase the previous solution See the Glossary Repeatedly calling fit or partialfit when warmstart is True can result in a different solution than when calling fit a single time because of the way the data is shuffled

classweight dict classlabel weight or “balanced” or None optional Preset for the classweight fit parameter

Weights associated with classes If not given all classes are supposed to have weight one The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as nsamples nclasses np

bincounty

New in version 017 parameter classweight to automatically weight samples

average bool or int optional When set to True computes the averaged SGD weights and stores the result in the coef attribute If set to an int greater than 1 averaging will begin once the total number of samples seen reaches average So average10 will begin averaging after seeing 10 samples

New in version 019 parameter average to use weights averaging in SGD

Attributes

coef array shape 1 nfeatures if nclasses 2 else nclasses nfeatures Weights assigned to the features

intercept array shape 1 if nclasses 2 else nclasses Constants in decision function

niter int The actual number of iterations to reach the stopping criterion For multiclass fits it is the maximum over every binary fit

See also

SGDClassifier

Perceptron

1916 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

References

Online PassiveAggressive Algorithms <http://jmlr.csail.mit.edu/papers/volume7/crammer06a/crammer06a.pdf> K Crammer O Dekel J Keshat S ShalevShwartz Y Singer JMLR 2006

Examples

```
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.datasets import make_classification
X, y = make_classification(n_features=4, random_state=0)
clf = PassiveAggressiveClassifier(max_iter=1000, random_state=0,
tol=1e-3,
clf_fit_X=y
PassiveAggressiveClassifier(C=10, average=False, class_weight=None,
early_stopping=False, fit_intercept=True, loss='hinge',
max_iter=1000, niternochange=5, n_jobs=None,
random_state=0, shuffle=True, tol=0.001,
validation_fraction=0.1, verbose=0, warm_start=False)
print(clf.coef_)
0.26642044 0.45070924 0.67251877 0.64185414
print(clf.intercept_)
1.84127814
print(clf.predict(0, 0, 0, 0))
1
```

Methods

decision_function(self, X) Predict confidence scores for samples
densify(self) Convert coefficient matrix to dense array format
fit(self, X, y, coef_init=None, intercept_init=None) Fit linear model with Passive Aggressive algorithm
get_params(self, deep=True) Get parameters for this estimator
partial_fit(self, X, y, classes=None) Fit linear model with Passive Aggressive algorithm
predict(self, X) Predict class labels for samples in X
score(self, X, y, sample_weight=None) Returns the mean accuracy on the given test data and labels
set_params(self, **kwargs) Set parameters for this estimator
sparsify(self) Convert coefficient matrix to sparse format
init(self, C=10, fit_intercept=True, max_iter=1000, tol=0.001, early_stopping=False, validation_fraction=0.1, niternochange=5, shuffle=True, verbose=0, loss='hinge', n_jobs=None, random_state=None, warm_start=False, class_weight=None, average=False)
decision_function(self, X) Predict confidence scores for samples
The confidence score for a sample is the signed distance of that sample to the hyperplane
Parameters
X: array-like or sparse matrix shape (n_samples, n_features) Samples
622sklearn.linear_model Generalized Linear Models 1917

scikitlearn user guide Release 0213

Returns

array shapensamples if nclasses > 2 else nsamples nclasses Confidence scores per sample class combination In the binary case confidence score for selfclasses1 where 0 means this class would be predicted

densifyself

Convert coefficient matrix to dense array format

Converts the coef member back to a numpyndarray This is the default format of coef and is required for fitting so calling this method is only required on models that have previously been sparsified otherwise it is a noop

Returns

self estimator

fitselfXycoefinitNone interceptinitNone

Fit linear model with Passive Aggressive algorithm

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data

ynumpy array of shape nsamples Target values

coefinit array shape nclassesnfeatures The initial coefficients to warmstart the optimization

interceptinit array shape nclasses The initial intercept to warmstart the optimization

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyclassesNone

Fit linear model with Passive Aggressive algorithm

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Subset of the training data

ynumpy array of shape nsamples Subset of the target values

classes array shape nclasses Classes across all calls to partialfit Can be obtained by vianpuniqueyall where yall is the target vector of the entire dataset This

argument is required for the first call to partialfit and can be omitted in the subsequent calls Note that y doesn't need to contain all labels in classes

Returns

self returns an instance of self

1918 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Predicted class label per sample

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

sparsify self

Convert coefficient matrix to sparse format

Converts the coef member to a scipysparse matrix which for L1regularized models can be much more memory and storageefficient than the usual numpyndarray representation

Theintercept member is not converted

Returns

self estimator

Notes

For nonsparse models ie when there are not many zeros in coef this may actually increase memory usage so use this method with care A rule of thumb is that the number of zero elements which can be computed with coef_0sum must be more than 50 for this to provide significant benefits

After calling this method further fitting with the partialfit method if any will not work until you call

densify

Examples using sklearnlinearmodelPassiveAggressiveClassifier

•Outofcore classification of text documents

•Comparing various online solvers

•Classification of text documents using sparse features

622sklearnlinearmodel Generalized Linear Models 1919

scikitlearn user guide Release 0.21.3

62214sklearnlinearmodel PassiveAggressiveRegressor

classsklearnlinearmodel PassiveAggressiveRegressor C10 fitintercept=True

maxiter=1000 tol=0.001

earlystopping=False

validationfraction=0.1

niternochange=5 shuffle

fit=True verbose=0

loss='epsiloninsensitive' epsilon

randomstate=None

warmstart=False average

age=False

Passive Aggressive Regressor

Read more in the User Guide

Parameters

Cfloat Maximum step size regularization Defaults to 10

fitintercept bool Whether the intercept should be estimated or not If False the data is assumed to be already centered Defaults to True

maxiter int optional default=1000 The maximum number of passes over the training data aka epochs It only impacts the behavior in the fit method and not the partialfit

New in version 0.19

tol float or None optional default=1e-3 The stopping criterion If it is not None the iterations will stop when loss - previousloss < tol

New in version 0.19

earlystopping bool default=False Whether to use early stopping to terminate training when validation score is not improving If set to True it will automatically set aside a fraction of training data as validation and terminate training when validation score is not improving by at least tol for niternochange consecutive epochs

New in version 0.20

validationfraction float default=0.1 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if earlystopping is True

New in version 0.20

niternochange int default=5 Number of iterations with no improvement to wait before early stopping

New in version 0.20

shuffle bool default=True Whether or not the training data should be shuffled after each epoch

verbose integer optional The verbosity level

loss string optional The loss function to be used epsiloninsensitive equivalent to PAI in the reference paper squaredepsiloninsensitive equivalent to PAII in the reference paper

epsilon float If the difference between the current prediction and the correct label is below this threshold the model is not updated

randomstate int RandomState instance or None optional default=None The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is

1920 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

the random number generator If None the random number generator is the RandomState instance used by nprandom

warmstart bool optional When set to True reuse the solution of the previous call to fit as initialization otherwise just erase the previous solution See the Glossary

Repeatedly calling fit or partialfit when warmstart is True can result in a different solution than when calling fit a single time because of the way the data is shuffled

average bool or int optional When set to True computes the averaged SGD weights and stores the result in the coef attribute If set to an int greater than 1 averaging will begin once the total number of samples seen reaches average So average10 will begin averaging after seeing 10 samples

New in version 019 parameter average to use weights averaging in SGD

Attributes

coef array shape 1 nfeatures if nclasses 2 else nclasses nfeatures Weights assigned to the features

intercept array shape 1 if nclasses 2 else nclasses Constants in decision function

niter int The actual number of iterations to reach the stopping criterion

See also

SGDRegressor

References

Online PassiveAggressive Algorithms <http://jmlr.csail.mit.edu/papers/volume7/crammer06a/crammer06a.pdf> K Crammer O Dekel J Keshat S ShalevShwartz Y Singer JMLR 2006

Examples

```
from sklearn.linear_model import PassiveAggressiveRegressor
from sklearn.datasets import make_regression
X, y = make_regression(n_features=4, random_state=0)
regr = PassiveAggressiveRegressor(max_iter=100, random_state=0,
tol=1e-3)
regr.fit(X, y)
PassiveAggressiveRegressor(C=10, average=False, early_stopping=False,
epsilon=0.1, fit_intercept=True, loss_epsilon_insensitive=True,
max_iter=100, n_iter_no_change=5, random_state=0,
shuffle=True, tol=0.001, validation_fraction=0.1,
verbose=0, warm_start=False)
print(regr.coef_)
[ 20.48736655  34.18818427  67.59122734  87.94731329]
print(regr.intercept_)
[ 0.2306214]
print(regr.predict([0, 0, 0, 0]))
[ 0.2306214]
622sklearn.linear_model Generalized Linear Models 1921
```

scikitlearn user guide Release 0213

Methods

densify self Convert coefficient matrix to dense array format

fitself X y coefinit interceptinit Fit linear model with Passive Aggressive algorithm

getparams self deep Get parameters for this estimator

partialfit self X y Fit linear model with Passive Aggressive algorithm

predict self X Predict using the linear model

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self args kwargs

sparsify self Convert coefficient matrix to sparse format

init selfC10 fitinterceptTrue maxiter1000 tol0001 earlystoppingFalse validationfraction01 niternochange5 shuffleTrue verbose0

loss'epsiloninsensitive' epsilon01 randomstateNone warmstartFalse averageFalse

densifyself

Convert coefficient matrix to dense array format

Converts the coef member back to a numpyndarray This is the default format of coef and is required for fitting so calling this method is only required on models that have previously been sparsified otherwise it is a noop

Returns

self estimator

fitselfXycoefinitNone interceptinitNone

Fit linear model with Passive Aggressive algorithm

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data

ynumpy array of shape nsamples Target values

coefinit array shape nfeatures The initial coefficients to warmstart the optimization

interceptinit array shape 1 The initial intercept to warmstart the optimization

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXy

Fit linear model with Passive Aggressive algorithm

Parameters

1922 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike sparse matrix shape nsamples nfeatures Subset of training data

ynumpy array of shape nsamples Subset of target values

Returns

self returns an instance of self

predictselfX

Predict using the linear model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures

Returns

array shape nsamples Predicted target values per element in X

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

sparsify self

Convert coefficient matrix to sparse format

Converts the coef member to a scipysparse matrix which for L1regularized models can be much more

memory and storageefficient than the usual numpyndarray representation

Theintercept member is not converted

Returns

self estimator

622sklearnlinearmodel Generalized Linear Models 1923

Notes

For nonsparse models ie when there are not many zeros in coef this may actually increase memory usage so use this method with care A rule of thumb is that the number of zero elements which can be computed with coef_0sum must be more than 50 for this to provide significant benefits After calling this method further fitting with the partialfit method if any will not work until you call densify

62215sklearnlinear Perceptron
class sklearnlinear Perceptron penalty=None alpha=0.0001 fit_intercept=True
max_iter=1000 tol=0.0001 shuffle=True verbose=0
eta=0.1 n_jobs=None random_state=0
early_stopping=False validation_fraction=0.1
n_iter_no_change=5 class_weight=None
warm_start=False

Read more in the User Guide

Parameters

penalty None 'l2' or 'l1' or 'elasticnet' The penalty aka regularization term to be used
Defaults to None
alpha float Constant that multiplies the regularization term if regularization is used Defaults to 0.0001
fit_intercept bool Whether the intercept should be estimated or not If False the data is assumed to be already centered Defaults to True
max_iter int optional default 1000 The maximum number of passes over the training data aka epochs It only impacts the behavior in the fit method and not the partialfit
New in version 0.19
tol float or None optional default 1e-3 The stopping criterion If it is not None the iterations will stop when loss - previous_loss > tol
New in version 0.19
shuffle bool optional default True Whether or not the training data should be shuffled after each epoch
verbose integer optional The verbosity level
eta double Constant by which the updates are multiplied Defaults to 1
n_jobs int or None optional default None The number of CPUs to use to do the O.V.A. One Versus All for multiclass problems computation None means 1 unless in a joblib parallel backend context 1 means using all processors See Glossary for more details
random_state int RandomState instance or None optional default None The seed of the pseudo random number generator to use when shuffling the data If int random_state is the seed used by the random number generator If RandomState instance random_state is the random number generator If None the random number generator is the RandomState instance used by np.random
early_stopping bool default False Whether to use early stopping to terminate training when validation score is not improving If set to True it will automatically set aside a stratified

1924 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

fraction of training data as validation and terminate training when validation score is not improving by at least tol for niternochange consecutive epochs

New in version 020

validationfraction float default01 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if earlystopping is True

New in version 020

niternochange int default5 Number of iterations with no improvement to wait before early stopping

New in version 020

classweight dict classlabel weight or “balanced” or None optional Preset for the classweight fit parameter

Weights associated with classes If not given all classes are supposed to have weight one

The “balanced” mode uses the values of y to automatically adjust weights inversely pro

portional to class frequencies in the input data as nsamples nclasses np

bincounty

warmstart bool optional When set to True reuse the solution of the previous call to fit as

initialization otherwise just erase the previous solution See the Glossary

Attributes

coef array shape 1 nfeatures if nclasses 2 else nclasses nfeatures Weights

assigned to the features

intercept array shape 1 if nclasses 2 else nclasses Constants in decision function

niter int The actual number of iterations to reach the stopping criterion For multiclass fits

it is the maximum over every binary fit

See also

SGDClassifier

Notes

Perceptron is a classification algorithm which shares the same underlying implementation with

SGDClassifier In factPerceptron is equivalent to SGDClassifierlossperceptron

eta01 learningrateconstant penaltyNone

References

<https://en.wikipedia.org/wiki/Perceptron> and references therein

Examples

from sklearn.datasets import load_digits

from sklearn.linear_model import Perceptron

X y load_digits.returnXy True

clf Perceptron(tol=1e-3 random_state=0

clffitX y

622sklearn.linear_model Generalized Linear Models 1925

scikitlearn user guide Release 0213

Perceptronalpha00001 classweightNone earlystoppingFalse eta010
fitinterceptTrue maxiter1000 niternochange5 njobsNone
penaltyNone randomstate0 shuffleTrue tol0001
validationfraction01 verbose0 warmstartFalse
clfscoreX y
0939

Methods

decisionfunction self X Predict confidence scores for samples

densify self Convert coefficient matrix to dense array format

fitself X y coefinit interceptinit Fit linear model with Stochastic Gradient Descent

getparams self deep Get parameters for this estimator

partialfit self X y classes sampleweight Perform one epoch of stochastic gradient descent on given samples

predict self X Predict class labels for samples in X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self args kwargs

sparsify self Convert coefficient matrix to sparse format

init selfpenaltyNone alpha00001 fitinterceptTrue maxiter1000 tol0001 shuffleTrue verbose0 eta010 njobsNone randomstate0 earlystoppingFalse validationfraction01 niternochange5 classweightNone warmstartFalse

decisionfunction selfX

Predict confidence scores for samples

The confidence score for a sample is the signed distance of that sample to the hyperplane

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

array shapensamples if nclasses > 2 else nsamples nclasses Confidence scores per sample class combination In the binary case confidence score for selfclasses1 where 0 means this class would be predicted

densifyself

Convert coefficient matrix to dense array format

Converts the coef member back to a numpyndarray This is the default format of coef and is required for fitting so calling this method is only required on models that have previously been sparsified otherwise it is a noop

Returns

self estimator

fitselfXycoefinitNone interceptinitNone sampleweightNone

Fit linear model with Stochastic Gradient Descent

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data

1926 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

ynumpy array shape nsamples Target values

coefinit array shape nclasses nfeatures The initial coefficients to warmstart the optimization

interceptinit array shape nclasses The initial intercept to warmstart the optimization

sampleweight arraylike shape nsamples optional Weights applied to individual samples If not provided uniform weights are assumed These weights will be multiplied with classweight passed through the constructor if classweight is specified

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyclassesNone sampleweightNone

Perform one epoch of stochastic gradient descent on given samples

Internally this method uses maxiter 1 Therefore it is not guaranteed that a minimum of the cost

function is reached after calling it once Matters such as objective convergence and early stopping should be handled by the user

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Subset of the training data

ynumpy array shape nsamples Subset of the target values

classes array shape nclasses Classes across all calls to partialfit Can be obtained by

vianpuniqueyall where yall is the target vector of the entire dataset This

argument is required for the first call to partialfit and can be omitted in the subsequent

calls Note that y doesn't need to contain all labels in classes

sampleweight arraylike shape nsamples optional Weights applied to individual

samples If not provided uniform weights are assumed

Returns

self returns an instance of self

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Predicted class label per sample

622sklearnlinearmodel Generalized Linear Models 1927

scikitlearn user guide Release 0213

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

sparsify self

Convert coefficient matrix to sparse format

Converts the coef member to a scipysparse matrix which for L1regularized models can be much more memory and storageefficient than the usual numpyndarray representation

Theintercept member is not converted

Returns

self estimator

Notes

For nonsparse models ie when there are not many zeros in coef this may actually increase memory usage so use this method with care A rule of thumb is that the number of zero elements which can be computed with coef_0sum must be more than 50 for this to provide significant benefits

After calling this method further fitting with the partialfit method if any will not work until you call

densify

Examples using sklearnlinearmodelPerceptron

•Outofcore classification of text documents

•Comparing various online solvers

•Classification of text documents using sparse features

62216sklearnlinearmodel RANSACRegressor

classsklearnlinearmodel RANSACRegressor baseestimatorNone minsamplesNone

residualthresholdNone isdatavalidNone

ismodelvalidNone maxtrials100

maxskipsinf stopninliersinf

stopscoreinf stopprobability099

loss'absoluteloss' randomstateNone

RANSAC RANdom SAmple Consensus algorithm

RANSAC is an iterative algorithm for the robust estimation of parameters from a subset of inliers from the complete data set More information can be found in the general documentation of linear models

1928 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

A detailed description of the algorithm can be found in the documentation of the linearmodel subpackage
Read more in the User Guide

Parameters

baseestimator object optional Base estimator object which implements the following methods

- fitX y Fit model to given training data and target values
- scoreX y Returns the mean accuracy on the given test data which is used for the stop criterion defined by stopscore Additionally the score is used to decide which of two equally large consensus sets is chosen as the better one
- predictX Returns predicted values using the linear model which is used to compute residual error using loss function

If baseestimator is None then baseestimatorsklearnlinearmodel

LinearRegression is used for target values of dtype float

Note that the current implementation only supports regression estimators

minsamples int 1 or float 0 1 optional Minimum number of samples chosen

randomly from original data Treated as an absolute number of samples for

minsamples 1 treated as a relative number ceilminsamples X

shape0 forminsamples 1 This is typically chosen as the minimal number

of samples necessary to estimate the given baseestimator By default a sklearn

linearmodelLinearRegression estimator is assumed and minsamples is

chosen asXshape1 1

residualthreshold float optional Maximum residual for a data sample to be classified as an inlier By default the threshold is chosen as the MAD median absolute deviation of the target values y

isdatavalid callable optional This function is called with the randomly selected data before the model is fitted to it isdatavalidX y If its return value is False the current randomly chosen subsample is skipped

ismodelvalid callable optional This function is called with the estimated model and the randomly selected data ismodelvalidmodel X y If its return value is False the current randomly chosen subsample is skipped Rejecting samples with this function is computationally costlier than with isdatavalid ismodelvalid should therefore only be used if the estimated model is needed for making the rejection decision
maxtrials int optional Maximum number of iterations for random sample selection
maxskips int optional Maximum number of iterations that can be skipped due to finding zero inliers or invalid data defined by isdatavalid or invalid models defined by ismodelvalid

New in version 019

stopninliers int optional Stop iteration if at least this number of inliers are found

stopscore float optional Stop iteration if score is greater equal than this threshold

stopprobability float in range 0 1 optional RANSAC iteration stops if at least one outlier free set of the training data is sampled in RANSAC This requires to generate at least N samples iterations

N log1 probability log1 e m

622sklearnlinearmodel Generalized Linear Models 1929

scikitlearn user guide Release 0213

where the probability confidence is typically set to high value such as 0.99 the default
and e is the current fraction of inliers wrt the total number of samples
loss string callable optional default "absoluteloss" String inputs "absoluteloss" and
"squaredloss" are supported which find the absolute loss and squared loss per sample re
spectively
If `floss` is a callable then it should be a function that takes two arrays as inputs the true
and predicted value and returns a 1D array with the i th value of the array corresponding to
the loss on X_i
If the loss on a sample is greater than the `residualthreshold` then this sample is
classified as an outlier
`randomstate` int RandomState instance or None optional default None The generator used
to initialize the centers If int `randomstate` is the seed used by the random number gener
ator If RandomState instance `randomstate` is the random number generator If None the
random number generator is the RandomState instance used by `nprandom`
Attributes
`estimator` object Best fitted model copy of the `baseestimator` object
`ntrials` int Number of random selection trials until one of the stop criteria is met It is
always `maxtrials`
`inliermask` bool array of shape `nsamples` Boolean mask of inliers classified as True
`nskipsnotinliers` int Number of iterations skipped due to finding zero inliers
New in version 0.19
`nskipsinvaliddata` int Number of iterations skipped due to invalid data defined by
`isdatavalid`
New in version 0.19
`nskipsinvalidmodel` int Number of iterations skipped due to an invalid model defined by
`ismodelvalid`
New in version 0.19
References
R80ce5b25cf9d1 R80ce5b25cf9d2 R80ce5b25cf9d3
Examples
from sklearn.linear_model import RANSACRegressor
from sklearn.datasets import make_regression
`X, y` = make_regression
`nsamples=200 nfeatures=2 noise=40 randomstate=0`
`reg = RANSACRegressor(randomstate=0)`
`fitX, y`
`reg.score(X, y)`
0.9885
`reg.predict(X)`
array([31.9417])
1930 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Methods

fitself X y sampleweight Fit estimator using RANSAC algorithm

getparams self deep Get parameters for this estimator

predict self X Predict using the estimated model

score self X y Returns the score of the prediction

setparams self params Set the parameters of this estimator

init self baseestimatorNone minsamplesNone residualthresholdNone

isdatavalidNone ismodelvalidNone maxtrials100 maxskipsinf

stopninliersinf stopscoreinf stopprobability099 loss'absoluteloss' ran

domstateNone

fitselfXysampleweightNone

Fit estimator using RANSAC algorithm

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Training data

yarraylike shape nsamples or nsamples ntargets Target values

sampleweight arraylike shape nsamples Individual weights for each sample raises error if sampleweight is passed and baseestimator fit method does not support it

Raises

ValueError If no valid consensus set could be found This occurs if isdatavalid and

ismodelvalid return False for all maxtrials randomly chosen subsamples

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the estimated model

This is a wrapper for estimatorpredictX

Parameters

Xnumpy array of shape nsamples nfeatures

Returns

yarray shape nsamples or nsamples ntargets Returns predicted values

scoreselfXy

Returns the score of the prediction

This is a wrapper for estimatorscoreX y

Parameters

Xnumpy array or sparse matrix of shape nsamples nfeatures Training data

622sklearnlinearmodel Generalized Linear Models 1931

scikitlearn user guide Release 0213

yarray shape nsamples or nsamples ntargets Target values

Returns

zfloat Score of the prediction

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnlinearmodelRANSACRegressor

- Robust linear model estimation using RANSAC

- TheilSen Regression

- Robust linear estimator fitting

62217sklearnlinearmodel Ridge

classsklearnlinearmodel Ridgealpha10 fitinterceptTrue normalizeFalse

copyXTrue maxiterNone tol0001 solver'auto'

randomstateNone

Linear least squares with l2 regularization

Minimizes the objective function

y Xw22 alpha w22

This model solves a regression model where the loss function is the linear least squares function and regularization is given by the l2norm Also known as Ridge Regression or Tikhonov regularization This estimator has builtin support for multivariate regression ie when y is a 2darray of shape nsamples ntargets

Read more in the User Guide

Parameters

alpha float arraylike shape ntargets Regularization strength must be a positive float

Regularization improves the conditioning of the problem and reduces the variance of the estimates Larger values specify stronger regularization Alpha corresponds to C1 in other

linear models such as LogisticRegression or LinearSVC If an array is passed penalties are

assumed to be specific to the targets Hence they must correspond in number

fitintercept boolean Whether to calculate the intercept for this model If set to false no

intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized be

fore regression by subtracting the mean and dividing by the l2norm If you wish to

standardize please use sklearnpreprocessingStandardScaler before

callingfit on an estimator with normalizeFalse

copyX boolean optional default True If True X will be copied else it may be overwritten

1932 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

maxiter int optional Maximum number of iterations for conjugate gradient solver For 'sparsecg' and 'lsqr' solvers the default value is determined by scipysparselinalg For 'sag' solver the default value is 1000

tolfloat Precision of the solution

solver 'auto' 'svd' 'cholesky' 'lsqr' 'sparsecg' 'sag' 'saga' Solver to use in the computational routines

- 'auto' chooses the solver automatically based on the type of data
- 'svd' uses a Singular Value Decomposition of X to compute the Ridge coefficients More stable for singular matrices than 'cholesky'
- 'cholesky' uses the standard scipy.linalg.solve function to obtain a closedform solution
- 'sparsecg' uses the conjugate gradient solver as found in scipysparselinalg.cg As an iterative algorithm this solver is more appropriate than 'cholesky' for largescale data possibility to set tol andmaxiter
- 'lsqr' uses the dedicated regularized leastsquares routine scipysparselinalg.lsqr It is the fastest and uses an iterative procedure
- 'sag' uses a Stochastic Average Gradient descent and 'saga' uses its improved unbiased version named SAGA Both methods also use an iterative procedure and are often faster than other solvers when both nsamples and nfeatures are large Note that 'sag' and 'saga' fast convergence is only guaranteed on features with approximately the same scale You can preprocess the data with a scaler from sklearn.preprocessing

All last five solvers support both dense and sparse data However only 'sag' and 'sparsecg' supports sparse input when fitintercept is True

New in version 017 Stochastic Average Gradient descent solver

New in version 019 SAGA solver

randomstate int RandomState instance or None optional default None The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random Used when solver 'sag'

New in version 017 randomstate to support Stochastic Average Gradient

Attributes

coef array shape nfeatures or ntargets nfeatures Weight vectors

intercept float array shape ntargets Independent term in decision function Set to 00

ifitintercept False

niter array or None shape ntargets Actual number of iterations for each target Avail

able only for sag and lsqr solvers Other solvers will return None

New in version 017

See also

RidgeClassifier Ridge classifier

RidgeCV Ridge regression with builtin cross validation

sklearn.kernelridge.KernelRidge Kernel ridge regression combines ridge regression with the kernel trick

622sklearn.linear_model Generalized Linear Models 1933

scikitlearn user guide Release 0213

Examples

```
from sklearn.linear_model import Ridge
import numpy as np
nsamples, nfeatures = 10, 5
rng = np.random.RandomState(0)
y = rng.randn(nsamples)
X = rng.randn(nsamples, nfeatures)
clf = Ridge(alpha=10)
clf.fit(X, y)
Ridge(alpha=10, copy_X=True, fit_intercept=True, max_iter=None,
normalize=False, random_state=None, solver='auto', tol=0.001)
```

Methods

```
fit(self, X, y, sample_weight) Fit Ridge regression model
get_params(self, deep) Get parameters for this estimator
predict(self, X) Predict using the linear model
score(self, X, y, sample_weight) Returns the coefficient of determination R2 of the prediction
set_params(self, **params) Set the parameters of this estimator
init(self, alpha=10, fit_intercept=True, normalize=False, copy_X=True, max_iter=None,
tol=0.001, solver='auto', random_state=None)
fit(self, X, y, sample_weight=None)
```

Fit Ridge regression model

Parameters

Xarraylike sparse matrix shape (nsamples, nfeatures) Training data
yarraylike shape (nsamples, or nsamples, ntargets) Target values
sample_weight float or numpy array of shape (nsamples,) Individual weights for each sample
Returns

self returns an instance of self
get_params(self, deep=True)
Get parameters for this estimator
Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators
Returns

params mapping of string to any Parameter names mapped to their values
predict(self, X)
Predict using the linear model
Parameters

Xarraylike or sparse matrix shape (nsamples, nfeatures) Samples
1934 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnlinearmodelRidge

•Compressive sensing tomography reconstruction with L1 prior Lasso

•Prediction Latency

•Plot Ridge coefficients as a function of the regularization

•Ordinary Least Squares and Ridge Regression Variance

•Plot Ridge coefficients as a function of the L2 regularization

•Polynomial interpolation

•HuberRegressor vs Ridge on dataset with strong outliers

622sklearnlinearmodel Generalized Linear Models 1935

scikitlearn user guide Release 0213

62218sklearnlinearmodel RidgeClassifier

classsklearnlinearmodel RidgeClassifier alpha10 fitinterceptTrue normalizeFalse

copyXTrue maxiterNone tol0001

classweightNone solver'auto' ran

domstateNone

Classifier using Ridge regression

Read more in the User Guide

Parameters

alpha float Regularization strength must be a positive float Regularization improves the conditioning of the problem and reduces the variance of the estimates Larger values specify stronger regularization Alpha corresponds to C1 in other linear models such as LogisticRegression or LinearSVC

fitintercept boolean Whether to calculate the intercept for this model If set to false no intercept will be used in calculations eg data is expected to be already centered

normalize boolean optional default False This parameter is ignored when

fitintercept is set to False If True the regressors X will be normalized before

regression by subtracting the mean and dividing by the l2norm If you wish to standardize please use sklearnpreprocessingStandardScaler before

callingfit on an estimator with normalizeFalse

copyX boolean optional default True If True X will be copied else it may be overwritten

maxiter int optional Maximum number of iterations for conjugate gradient solver The default value is determined by scipysparselinalg

tolfloat Precision of the solution

classweight dict or 'balanced' optional Weights associated with classes in the form

classlabel weight If not given all classes are supposed to have weight one

The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as nsamples nclasses np

bincounty

solver 'auto' 'svd' 'cholesky' 'lsqr' 'sparsecg' 'sag' 'saga' Solver to use in the computational routines

- 'auto' chooses the solver automatically based on the type of data
 - 'svd' uses a Singular Value Decomposition of X to compute the Ridge coefficients More stable for singular matrices than 'cholesky'
 - 'cholesky' uses the standard scipy.linalg.solve function to obtain a closedform solution
 - 'sparsecg' uses the conjugate gradient solver as found in scipysparselinalgcg As an iterative algorithm this solver is more appropriate than 'cholesky' for largescale data possibility to set tol andmaxiter
 - 'lsqr' uses the dedicated regularized leastsquares routine scipysparselinalglsqr It is the fastest and uses an iterative procedure
 - 'sag' uses a Stochastic Average Gradient descent and 'saga' uses its unbiased and more flexible version named SAGA Both methods use an iterative procedure and are often faster than other solvers when both nsamples and nfeatures are large Note that 'sag' and 'saga' fast convergence is only guaranteed on features with approximately the same scale You can preprocess the data with a scaler from sklearnpreprocessing
- 1936 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

New in version 017 Stochastic Average Gradient descent solver

New in version 019 SAGA solver

randomstate int RandomState instance or None optional default None The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom Used when solver 'sag'

Attributes

coef array shape 1 nfeatures or nclasses nfeatures Coefficient of the features in the decision function

coef is of shape 1 nfeatures when the given problem is binary

intercept float array shape ntargets Independent term in decision function Set to 00

ffitintercept False

niter array or None shape ntargets Actual number of iterations for each target Avail

able only for sag and lsqr solvers Other solvers will return None

See also

Ridge Ridge regression

RidgeClassifierCV Ridge classifier with builtin cross validation

Notes

For multiclass classification nclass classifiers are trained in a oneversusall approach Concretely this is implemented by taking advantage of the multivariate response support in Ridge

Examples

```
from sklearndatasets import loadbreastcancer
from sklearnlinearmodel import RidgeClassifier
```

```
X y loadbreastcancerreturnXy True
```

```
clf RidgeClassifierfitX y
```

```
clf.scoreX y
```

```
09595
```

Methods

decisionfunction self X Predict confidence scores for samples

fitself X y sampleweight Fit Ridge regression model

getparams self deep Get parameters for this estimator

predict self X Predict class labels for samples in X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

622sklearnlinearmodel Generalized Linear Models 1937

scikitlearn user guide Release 0213

init selfalpha10 fitinterceptTrue normalizeFalse copyXTrue maxiterNone

tol0001 classweightNone solver'auto' randomstateNone

decisionfunction selfX

Predict confidence scores for samples

The confidence score for a sample is the signed distance of that sample to the hyperplane

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

array shapensamples if nclasses > 2 else nsamples nclasses Confidence

scores per sample class combination In the binary case confidence score for

selfclasses1 where 0 means this class would be predicted

fitselfXysampleweightNone

Fit Ridge regression model

Parameters

Xarraylike sparse matrix shape nsamplesnfeatures Training data

yarraylike shape nsamples Target values

sampleweight float or numpy array of shape nsamples Sample weight

New in version 017 sampleweight support to Classifier

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Predicted class label per sample

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each

sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

1938 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnlinearmodelRidgeClassifier

•Classification of text documents using sparse features

62219sklearnlinearmodel SGDClassifier

classsklearnlinearmodel SGDClassifier loss'hinge' penalty'l2' alpha00001

l1ratio015 fitinterceptTrue maxiter1000

tol0001 shuffleTrue verbose0 epsilon01

njobsNone randomstateNone learn

ingrate'optimal' eta000 powert05

earlystoppingFalse validationfraction01

niternochange5 classweightNone

warmstartFalse averageFalse

Linear classifiers SVM (Support Vector Machine) logistic regression ao with SGD training

This estimator implements regularized linear models with stochastic gradient descent SGD learning the gra

dient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing

strength schedule aka learning rate SGD allows minibatch onlineoutofcore learning see the partialfit

method For best results using the default learning rate schedule the data should have zero mean and unit

variance

This implementation works with data represented as dense or sparse arrays of floating point values for the

features The model it fits can be controlled with the loss parameter by default it fits a linear support vector

machine SVM (Support Vector Machine)

The regularizer is a penalty added to the loss function that shrinks model parameters towards the zero vector

using either the squared euclidean norm L2 or the absolute norm L1 or a combination of both Elastic Net If

the parameter update crosses the 00 value because of the regularizer the update is truncated to 00 to allow for

learning sparse models and achieve online feature selection

Read more in the User Guide

Parameters

loss str default 'hinge' The loss function to be used Defaults to 'hinge' which gives a linear

SVM (Support Vector Machine)

The possible options are 'hinge' 'log' 'modifiedhuber' 'squaredhinge' 'per

ceptron' or a regression loss 'squaredloss' 'huber' 'epsiloninsensitive' or

'squaredepsiloninsensitive'

622sklearnlinearmodel Generalized Linear Models 1939

scikitlearn user guide Release 0213

The 'log' loss gives logistic regression a probabilistic classifier 'modifiedhuber' is another smooth loss that brings tolerance to outliers as well as probability estimates 'squaredhinge' is like hinge but is quadratically penalized 'perceptron' is the linear loss used by the perceptron algorithm The other losses are designed for regression but can be useful in classification as well see SGDRegressor for a description

penalty str 'none' 'l2' 'l1' or 'elasticnet' The penalty aka regularization term to be used Defaults to 'l2' which is the standard regularizer for linear SVM (Support Vector Machine) models 'l1' and 'elasticnet' might bring sparsity to the model feature selection not achievable with 'l2'

alpha float Constant that multiplies the regularization term Defaults to 0.0001 Also used to compute learningrate when set to 'optimal'

l1ratio float The Elastic Net mixing parameter with 0 l1ratio 1 l1ratio0 corresponds to L2 penalty l1ratio1 to L1 Defaults to 0.15

fitintercept bool Whether the intercept should be estimated or not If False the data is assumed to be already centered Defaults to True

maxiter int optional default1000 The maximum number of passes over the training data aka epochs It only impacts the behavior in the fit method and not the partialfit

New in version 0.19

tol float or None optional default1e3 The stopping criterion If it is not None the iterations will stop when loss - bestloss > tol for niter no change consecutive epochs

New in version 0.19

shuffle bool optional Whether or not the training data should be shuffled after each epoch Defaults to True

verbose integer optional The verbosity level

epsilon float Epsilon in the epsilon insensitive loss functions only if loss is 'huber' 'epsilon insensitive' or 'squared epsilon insensitive' For 'huber' determines the threshold at which it becomes less important to get the prediction exactly right For epsilon insensitive any differences between the current prediction and the correct label are ignored if they are less than this threshold

njobs int or None optional defaultNone The number of CPUs to use to do the OVS A One Versus All for multiclass problems computation None means 1 unless in a joblib parallel backend context 1 means using all processors See Glossary for more details

randomstate int RandomState instance or None optional defaultNone The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random

learningrate string optional The learning rate schedule

'constant' eta eta0

'optimal' default eta = 10 * alpha * t / t0 where t0 is chosen by a heuristic proposed by Leon Bottou

'invscaling' eta eta0 * pow(t, -power)

'adaptive' eta eta0 as long as the training keeps decreasing Each time

niter no change consecutive epochs fail to decrease the training loss by tol or fail to

1940 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0.21.3

increase validation score by tol if earlystopping is True the current learning rate is divided by 5

eta0 double The initial learning rate for the ‘constant’ ‘invscaling’ or ‘adaptive’ schedules The default value is 0.0 as eta0 is not used by the default schedule ‘optimal’

power_t double The exponent for inverse scaling learning rate default 0.5

earlystopping bool default False Whether to use early stopping to terminate training when validation score is not improving If set to True it will automatically set aside a stratified fraction of training data as validation and terminate training when validation score is not improving by at least tol for niter_nocchange consecutive epochs

New in version 0.20

validationfraction float default 0.1 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if earlystopping is True

New in version 0.20

niter_nocchange int default 5 Number of iterations with no improvement to wait before early stopping

New in version 0.20

class_weight dict class_label weight or “balanced” or None optional Preset for the class_weight fit parameter

Weights associated with classes If not given all classes are supposed to have weight one The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / n_classes np

bincount_y warmstart bool optional When set to True reuse the solution of the previous call to fit as initialization otherwise just erase the previous solution See the Glossary Repeatedly calling fit or partial_fit when warmstart is True can result in a different solution than when calling fit a single time because of the way the data is shuffled If a dynamic learning rate is used the learning rate is adapted depending on the number of samples already seen Calling fit resets this counter while partial_fit will result in increasing the existing counter

average bool or int optional When set to True computes the averaged SGD weights and stores the result in the coef attribute If set to an int greater than 1 averaging will begin once the total number of samples seen reaches average So average=10 will begin averaging after seeing 10 samples

Attributes

coef array shape (1, n_features) if n_classes == 2 else (n_classes, n_features) Weights assigned to the features

intercept array shape (1,) if n_classes == 2 else (n_classes,) Constants in decision function

n_iter int The actual number of iterations to reach the stopping criterion For multiclass fits it is the maximum over every binary fit

loss_function concrete LossFunction

See also

sklearnsvm.LinearSVC LogisticRegression Perceptron

622sklearn.linear_model.Generalized Linear Models 1941

scikitlearn user guide Release 0213

Examples

```
import numpy as np
from sklearn import linear_model
X = np.array([1, 2, 1, 1, 1, 2, 1])
Y = np.array([1, 2, 2])
clf = linear_model.SGDClassifier(max_iter=1000, tol=1e-3)
clf.fit(X, Y)
```

```
SGDClassifier(alpha=0.0001, average=False, class_weight=None,
early_stopping=False, epsilon=0.1, eta=0.00, fit_intercept=True,
l1_ratio=0.15, learning_rate='optimal', loss='hinge', max_iter=1000,
n_iter_no_change=5, n_jobs=None, penalty='l2', power_t=0.5,
random_state=None, shuffle=True, tol=0.0001, validation_fraction=0.1,
verbose=0, warm_start=False)
print(clf.predict([0.8, 1.1]))
```

Methods

decision_function(self, X) Predict confidence scores for samples

densify(self) Convert coefficient matrix to dense array format

fit(self, X, y, coef_init=None, intercept_init=None) Fit linear model with Stochastic Gradient Descent

get_params(self, deep=True) Get parameters for this estimator

partial_fit(self, X, y, classes=None, sample_weight=None) Perform one epoch of stochastic gradient descent on given samples

predict(self, X) Predict class labels for samples in X

score(self, X, y, sample_weight=None) Returns the mean accuracy on the given test data and labels

set_params(self, **kwargs) Set parameters

sparsify(self) Convert coefficient matrix to sparse format

init(self, loss='hinge', penalty='l2', alpha=0.0001, l1_ratio=0.15, fit_intercept=True, max_iter=1000, tol=0.0001, shuffle=True, verbose=0, epsilon=0.1, n_jobs=None, random_state=None, learning_rate='optimal', eta=0.00, power_t=0.5, early_stopping=False, validation_fraction=0.1, n_iter_no_change=5, class_weight=None, warm_start=False, average=False)

decision_function(self, X) Predict confidence scores for samples

The confidence score for a sample is the signed distance of that sample to the hyperplane

Parameters

X: array-like or sparse matrix shape (n_samples, n_features) Samples

Returns

array shape (n_samples, n_classes) if n_classes > 2 else (n_samples, n_classes) Confidence scores per sample class combination In the binary case confidence score for self.classes_[1] where 0 means this class would be predicted

1942 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

densifyself

Convert coefficient matrix to dense array format

Converts the coef member back to a numpyndarray This is the default format of coef and is required for fitting so calling this method is only required on models that have previously been sparsified otherwise it is a noop

Returns

self estimator

fitselfXycoefinitNone interceptinitNone sampleweightNone

Fit linear model with Stochastic Gradient Descent

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data

ynumpy array shape nsamples Target values

coefinit array shape nclasses nfeatures The initial coefficients to warmstart the optimization

interceptinit array shape nclasses The initial intercept to warmstart the optimization

sampleweight arraylike shape nsamples optional Weights applied to individual samples If not provided uniform weights are assumed These weights will be multiplied with classweight passed through the constructor if classweight is specified

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyclassesNone sampleweightNone

Perform one epoch of stochastic gradient descent on given samples

Internally this method uses maxiter 1 Therefore it is not guaranteed that a minimum of the cost function is reached after calling it once Matters such as objective convergence and early stopping should be handled by the user

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Subset of the training data

ynumpy array shape nsamples Subset of the target values

classes array shape nclasses Classes across all calls to partialfit Can be obtained by vianpuniqueyall where yall is the target vector of the entire dataset This

argument is required for the first call to partialfit and can be omitted in the subsequent calls Note that y doesn't need to contain all labels in classes

sampleweight arraylike shape nsamples optional Weights applied to individual samples If not provided uniform weights are assumed

622sklearnlinearmodel Generalized Linear Models 1943

scikitlearn user guide Release 0213

Returns

self returns an instance of self

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Predicted class label per sample

predictlogproba

Log of probability estimates

This method is only available for log loss and modified Huber loss

When loss"modifiedhuber" probability estimates may be hard zeros and ones so taking the logarithm is not possible

Seepredictproba for details

Parameters

Xarraylike shape nsamples nfeatures

Returns

Tarraylike shape nsamples nclasses Returns the logprobability of the sample for each class in the model where classes are ordered as they are in selfclasses

predictproba

Probability estimates

This method is only available for log loss and modified Huber loss

Multiclass probability estimates are derived from binary onevsrest estimates by simple normalization as recommended by Zadrozny and Elkan

Binary probability estimates for loss"modifiedhuber" are given by clipdecisionfunctionX 1 1

1 2 For other loss functions it is necessary to perform proper probability calibration by wrapping the classifier with sklearnCalibrationCalibratedClassifierCV instead

Parameters

Xarraylike sparse matrix shape nsamples nfeatures

Returns

array shape nsamples nclasses Returns the probability of the sample for each class in the model where classes are ordered as they are in selfclasses

References

Zadrozny and Elkan "Transforming classifier scores into multiclass probability estimates" SIGKDD'02 <http://www.research.ibm.com/people/zadrozny/kdd2002/Transfpdf>

The justification for the formula in the loss"modifiedhuber" case is in the appendix B in <http://jmlr.csail.mit.edu/papers/volume2/zhang02c/zhang02c.pdf>

1944 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

sparsify self

Convert coefficient matrix to sparse format

Converts the coef member to a scipysparse matrix which for L1regularized models can be much more memory and storageefficient than the usual numpyndarray representation

Theintercept member is not converted

Returns

self estimator

Notes

For nonsparse models ie when there are not many zeros in coef this may actually increase memory usage so use this method with care A rule of thumb is that the number of zero elements which can be computed with coef_0sum must be more than 50 for this to provide significant benefits

After calling this method further fitting with the partialfit method if any will not work until you call

densify

Examples using sklearnlinearmodelSGDClassifier

- Model Complexity Influence
 - Outofcore classification of text documents
 - Pipelining chaining a PCA (Principal Component Analysis) and a logistic regression
 - SGD Maximum margin separating hyperplane
 - SGD Weighted samples
 - Comparing various online solvers
 - Plot multiclass SGD on the iris dataset
 - Early stopping of Stochastic Gradient Descent
 - Sample pipeline for text feature extraction and evaluation
 - Classification of text documents using sparse features
- 622sklearnlinearmodel Generalized Linear Models 1945

scikitlearn user guide Release 0213

```
62220sklearnlinearmodel SGDRegressor
classsklearnlinearmodel SGDRegressor loss'squaredloss' penalty'l2' alpha00001
l1ratio015 fitinterceptTrue maxiter1000
tol0001 shuffleTrue verbose0 epsilon01
randomstateNone learningrate'invscaling'
eta0001 powert025 earlystoppingFalse
validationfraction01 niternochange5
warmstartFalse averageFalse
```

Linear model fitted by minimizing a regularized empirical loss with SGD

SGD stands for Stochastic Gradient Descent the gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule aka learning rate

The regularizer is a penalty added to the loss function that shrinks model parameters towards the zero vector

using either the squared euclidean norm L2 or the absolute norm L1 or a combination of both Elastic Net If

the parameter update crosses the 00 value because of the regularizer the update is truncated to 00 to allow for learning sparse models and achieve online feature selection

This implementation works with data represented as dense numpy arrays of floating point values for the features

Read more in the User Guide

Parameters

loss str default 'squaredloss' The loss function to be used The possible values are

'squaredloss' 'huber' 'epsiloninsensitive' or 'squaredepsiloninsensitive'

The 'squaredloss' refers to the ordinary least squares fit 'huber' modifies 'squaredloss' to

focus less on getting outliers correct by switching from squared to linear loss past a distance

of epsilon 'epsiloninsensitive' ignores errors less than epsilon and is linear past that this

is the loss function used in SVR (Support Vector Regression) 'squaredepsiloninsensitive' is the same but becomes squared loss past a tolerance of epsilon

penalty str 'none' 'l2' 'l1' or 'elasticnet' The penalty aka regularization term to be used

Defaults to 'l2' which is the standard regularizer for linear SVM (Support Vector Machine) models 'l1' and 'elasticnet'

might bring sparsity to the model feature selection not achievable with 'l2'

alpha float Constant that multiplies the regularization term Defaults to 00001 Also used to

compute learningrate when set to 'optimal'

l1ratio float The Elastic Net mixing parameter with 0 l1ratio 1 l1ratio0 corre

sponds to L2 penalty l1ratio1 to L1 Defaults to 015

fitintercept bool Whether the intercept should be estimated or not If False the data is

assumed to be already centered Defaults to True

maxiter int optional default1000 The maximum number of passes over the training data

aka epochs It only impacts the behavior in the fit method and not the partialfit

New in version 019

tolfloat or None optional default1e3 The stopping criterion If it is not None the itera

tions will stop when loss bestloss tol for niternochange consecutive epochs

New in version 019

shuffle bool optional Whether or not the training data should be shuffled after each epoch

Defaults to True

verbose integer optional The verbosity level

1946 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

epsilon float Epsilon in the epsiloninsensitive loss functions only if loss is 'huber' 'epsiloninsensitive' or 'squaredepsiloninsensitive' For 'huber' determines the threshold at which it becomes less important to get the prediction exactly right For epsiloninsensitive any differences between the current prediction and the correct label are ignored if they are less than this threshold

randomstate int RandomState instance or None optional defaultNone The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

learningrate string optional The learning rate schedule

'constant' eta eta0

'optimal' eta 10 alpha t t0 where t0 is chosen by a heuristic proposed by Leon

Bottou

'invscaling' default eta eta0 powt powert

'adaptive' eta eta0 as long as the training keeps decreasing Each time

niternochange consecutive epochs fail to decrease the training loss by tol or fail to increase validation score by tol if earlystopping is True the current learning rate is divided by 5

eta0 double The initial learning rate for the 'constant' 'invscaling' or 'adaptive' schedules The default value is 001

powert double The exponent for inverse scaling learning rate default 05

earlystopping bool defaultFalse Whether to use early stopping to terminate training when validation score is not improving If set to True it will automatically set aside a fraction of training data as validation and terminate training when validation score is not improving by at least tol for niternochange consecutive epochs

New in version 020

validationfraction float default01 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if earlystopping is True

New in version 020

niternochange int default5 Number of iterations with no improvement to wait before early stopping

New in version 020

warmstart bool optional When set to True reuse the solution of the previous call to fit as initialization otherwise just erase the previous solution See the Glossary Repeatedly calling fit or partialfit when warmstart is True can result in a different solution than when calling fit a single time because of the way the data is shuffled If a dynamic learning rate is used the learning rate is adapted depending on the number of samples already seen Calling fit resets this counter while partialfit will result in increasing the existing counter

average bool or int optional When set to True computes the averaged SGD weights and stores the result in the coef attribute If set to an int greater than 1 averaging will begin once the total number of samples seen reaches average So average10 will begin averaging after seeing 10 samples

Attributes

622sklearnlinearmodel Generalized Linear Models 1947

scikitlearn user guide Release 0213

coef array shape nfeatures Weights assigned to the features
intercept array shape 1 The intercept term
averagecoef array shape nfeatures Averaged weights assigned to the features
averageintercept array shape 1 The averaged intercept term
niter int The actual number of iterations to reach the stopping criterion

See also
Ridge ElasticNet Lasso sklearnsvmSVR

Examples
import numpy as np
from sklearn import linearmodel
nsamples nfeatures 10 5
rng np.random.RandomState(0)
y rng.randn(nsamples)
X rng.randn(nsamples, nfeatures)
clf linearmodel.SGDRegressor(max_iter=1000, tol=1e-3)
clf.fit(X, y)

SGDRegressor(alpha=0.0001, average=False, early_stopping=False,
epsilon=0.1, eta=0.001, fit_intercept=True, l1_ratio=0.15,
learning_rate='invscaling', loss='squared_loss', max_iter=1000,
n_iter_no_change=5, penalty='l2', power_t=0.25, random_state=None,
shuffle=True, tol=0.0001, validation_fraction=0.1, verbose=0,
warm_start=False)
Methods
densify self Convert coefficient matrix to dense array format
fit self X y coef init intercept init Fit linear model with Stochastic Gradient Descent
get_params self deep Get parameters for this estimator
partial_fit self X y sample_weight Perform one epoch of stochastic gradient descent on
given samples
predict self X Predict using the linear model
score self X y sample_weight Returns the coefficient of determination R2 of the pre
diction
set_params self args kwargs
sparsify self Convert coefficient matrix to sparse format
init self loss='squared_loss' penalty='l2' alpha=0.0001 l1_ratio=0.15 fit_intercept=True
max_iter=1000 tol=0.0001 shuffle=True verbose=0 epsilon=0.1 random_state=None
learning_rate='invscaling' eta=0.001 power_t=0.25 early_stopping=False valida
tion_fraction=0.1 n_iter_no_change=5 warm_start=False average=False
densify self

Convert coefficient matrix to dense array format
Converts the coef member back to a numpy.ndarray This is the default format of coef and is
required for fitting so calling this method is only required on models that have previously been sparsified
1948 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

otherwise it is a noop

Returns

self estimator

fitselfXycoefinitNone interceptinitNone sampleweightNone

Fit linear model with Stochastic Gradient Descent

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training data

ynumpy array shape nsamples Target values

coefinit array shape nfeatures The initial coefficients to warmstart the optimization

interceptinit array shape 1 The initial intercept to warmstart the optimization

sampleweight arraylike shape nsamples optional Weights applied to individual

samples 1 for unweighted

Returns

self returns an instance of self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXysampleweightNone

Perform one epoch of stochastic gradient descent on given samples

Internally this method uses maxiter 1 Therefore it is not guaranteed that a minimum of the cost

function is reached after calling it once Matters such as objective convergence and early stopping should

be handled by the user

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Subset of training data

ynumpy array of shape nsamples Subset of target values

sampleweight arraylike shape nsamples optional Weights applied to individual

samples If not provided uniform weights are assumed

Returns

self returns an instance of self

predictselfX

Predict using the linear model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures

Returns

array shape nsamples Predicted target values per element in X

622sklearnlinearmodel Generalized Linear Models 1949

scikitlearn user guide Release 0213

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

sparsify self

Convert coefficient matrix to sparse format

Converts the coef member to a scipysparse matrix which for L1regularized models can be much more

memory and storageefficient than the usual numpyndarray representation

Theintercept member is not converted

Returns

self estimator

Notes

For nonsparse models ie when there are not many zeros in coef this may actually increase memory

usage so use this method with care A rule of thumb is that the number of zero elements which can be

computed with coef 0sum must be more than 50 for this to provide significant benefits

After calling this method further fitting with the partialfit method if any will not work until you call

densify

Examples using sklearnlinearmodelSGDRegressor

•Prediction Latency

1950 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

62221sklearnlinearmodel TheilSenRegressor

classsklearnlinearmodel TheilSenRegressor fitinterceptTrue copyXTrue

maxsubpopulation100000

nsubsamplesNone maxiter300

tol0001 randomstateNone

njobsNone verboseFalse

TheilSen Estimator robust multivariate regression model

The algorithm calculates least square solutions on subsets with size nsubsamples of the samples in X Any value of nsubsamples between the number of features and samples leads to an estimator with a compromise between robustness and efficiency Since the number of least square solutions is “nsamples choose nsubsamples” it can be extremely large and can therefore be limited with maxsubpopulation If this limit is reached the subsets are chosen randomly In a final step the spatial median or L1 median is calculated of all least square solutions
Read more in the User Guide

Parameters

fitintercept boolean optional default True Whether to calculate the intercept for this model
If set to false no intercept will be used in calculations

copyX boolean optional default True If True X will be copied else it may be overwritten
maxsubpopulation int optional default 1e4 Instead of computing with a set of cardinality ‘n choose k’ where n is the number of samples and k is the number of subsamples at least number of features consider only a stochastic subpopulation of a given maximal size if ‘n choose k’ is larger than maxsubpopulation For other than small problem sizes this parameter will determine memory usage and runtime if nsubsamples is not changed
nsubsamples int optional default None Number of samples to calculate the parameters
This is at least the number of features plus 1 if fitinterceptTrue and the number of samples as a maximum A lower number leads to a higher breakdown point and a low efficiency while a high number leads to a low breakdown point and a high efficiency If None take the minimum number of subsamples leading to maximal robustness If nsubsamples is set to nsamples TheilSen is identical to least squares

maxiter int optional default 300 Maximum number of iterations for the calculation of spatial median

tolfloat optional default 1e3 Tolerance when calculating spatial median

randomstate int RandomState instance or None optional default None A random number generator instance to define the state of the random permutations generator If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

njobs int or None optional defaultNone Number of CPUs to use during the cross validationNone means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

verbose boolean optional default False Verbose mode when fitting the model

Attributes

coef array shape nfeatures Coefficients of the regression model median of distribution

intercept float Estimated intercept of regression model

breakdown float Approximated breakdown point

niter int Number of iterations needed for the spatial median

622sklearnlinearmodel Generalized Linear Models 1951

scikitlearn user guide Release 0213

nsubpopulation int Number of combinations taken into account from ‘n choose k’ where
n is the number of samples and k is the number of subsamples

References

- TheilSen Estimators in a Multiple Linear Regression Model 2009 Xin Dang Hanxiang Peng Xueqin Wang and Heping Zhang <http://home.olemiss.edu/xiangdang/papers/MTSEpdf>

Examples

```
from sklearn.linear_model import TheilSenRegressor
from sklearn.datasets import make_regression
```

```
X, y = make_regression(
    n_samples=200, n_features=2, noise=40, random_state=0)
```

```
reg = TheilSenRegressor(random_state=0).fit(X, y)
```

```
reg.score(X, y)
```

```
0.9884
```

```
reg.predict(X)
```

```
array([315871])
```

Methods

fitself X y Fit linear model

getparams self deep Get parameters for this estimator

predict self X Predict using the linear model

score self X y sampleweight Returns the coefficient of determination R² of the prediction

setparams self params Set the parameters of this estimator

init self fit intercept True copy X True maxsubpopulation 100000 nsubsamples None

maxiter 300 tol 0.001 randomstate None njobs None verbose False

fitself X y

Fit linear model

Parameters

X numpy array of shape (n_samples, n_features) Training data

y numpy array of shape (n_samples,) Target values

Returns

self returns an instance of self

getparams self deep True

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

1952 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{true\text{mean}}$ 2sum The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 0.23 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnlinearmodelTheilSenRegressor

•TheilSen Regression

•Robust linear estimator fitting

622sklearnlinearmodel Generalized Linear Models 1953

scikitlearn user guide Release 0213

linearmodelenetpath X y l1ratio Compute elastic net path with coordinate descent

linearmodellarspath X y Xy Gram Compute Least Angle Regression or Lasso path using LARS algorithm 1

linearmodellarspathgram Xy Gram

nsampleslarspath in the sufficient stats mode 1

linearmodellassopath X y eps Compute Lasso path with coordinate descent

linearmodelorthogonalmp X y Orthogonal Matching Pursuit OMP

linearmodelorthogonalmpgram Gram Xy

Gram Orthogonal Matching Pursuit OMP

linearmodelridgeregression X y alpha

Solve the ridge equation by the method of normal equations

62222sklearnlinearmodel enetpath

sklearnlinearmodel enetpath Xyl1ratio05 eps0001 nalphas100 alphasNone

precompute'auto' XyNone copyXTrue coefinitNone

verboseFalse returnniterFalse positiveFalse

checkinputTrue params

Compute elastic net path with coordinate descent

The elastic net optimization function varies for mono and multioutputs

For monooutput tasks it is

1 2nsamples y Xw22

alpha l1ratio w1

05alpha1 l1ratio w22

For multioutput tasks it is

1 2 nsamples Y XWFro2

alpha l1ratio W21

05alpha1 l1ratio WFro2

Where

W21 sumi sqrtsumj wij2

ie the sum of norm of each row

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures Training data Pass directly as Fortran contiguous data to avoid unnecessary memory duplication If yis monooutput then Xcan be sparse

yndarray shape nsamples or nsamples noutputs Target values

l1ratio float optional float between 0 and 1 passed to elastic net scaling between l1 and l2

penaltiesl1ratio1 corresponds to the Lasso

eps float Length of the path eps1e3 means that alphamin alphamax 1e3

nalphas int optional Number of alphas along the regularization path

1954 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

alphas ndarray optional List of alphas where to compute the models If None alphas are set automatically

precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to speed up calculations If set to auto let us decide The Gram matrix can also be passed as argument

Xyarraylike optional Xy npdotXT y that can be precomputed It is useful only when the Gram matrix is precomputed

copyX boolean optional default True If True X will be copied else it may be overwritten

coefinit array shape nfeatures None The initial values of the coefficients

verbose bool or integer Amount of verbosity

returnniter bool whether to return the number of iterations or not

positive bool default False If set to True forces coefficients to be positive Only allowed whenydim 1

checkinput bool default True Skip input validation checks including the Gram matrix when provided assuming there are handled by the caller when checkinputFalse

params kwargs keyword arguments passed to the coordinate descent solver

Returns

alphas array shape nalphas The alphas along the path where models are computed

coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients along the path

dualgaps array shape nalphas The dual gaps at the end of the optimization for each alpha

niters arraylike shape nalphas The number of iterations taken by the coordinate descent optimizer to reach the specified tolerance for each alpha Is returned when returnniter is set to True

See also

MultiTaskElasticNet

MultiTaskElasticNetCV

ElasticNet

ElasticNetCV

Notes

For an example see `examples/linear_model/plot_lasso_coordinatedescent_path.py`

Examples using `sklearn.linear_model.enet_path`

•Lasso and Elastic Net

622sklearn.linear_model Generalized Linear Models 1955

scikitlearn user guide Release 0213

62223sklearnlinearmodel larspath

sklearnlinearmodel larspath XyXyNone GramNone maxiter500 alphamin0

method'lar' copyXTrue eps2220446049250313e

16copyGramTrue verbose0 returnpathTrue re

turnniterFalse positiveFalse

Compute Least Angle Regression or Lasso path using LARS algorithm 1

The optimization objective for the case method'lasso' is

1 2 nsamples y Xw22 alpha w1

in the case of method'lars' the objective function is only known in the form of an implicit equation see discussion in 1

Read more in the User Guide

Parameters

XNone or array shape nsamples nfeatures Input data Note that if X is None then the Gram matrix must be specified ie cannot be None or False

Deprecated since version 021 The use of XisNone in combination with Gram is not

None will be removed in v023 Use larspathgram instead

yNone or array shape nsamples Input targets

Xyarraylike shape nsamples or nsamples ntargets optional Xy npdotXT y that

can be precomputed It is useful only when the Gram matrix is precomputed

Gram None 'auto' array shape nfeatures nfeatures optional Precomputed Gram matrix

X' X if auto the Gram matrix is precomputed from the given X if there are more samples than features

Deprecated since version 021 The use of XisNone in combination with Gram is not None will be removed in v023 Use larspathgram instead

maxiter integer optional default500 Maximum number of iterations to perform set to infinity for no limit

alphamin float optional default0 Minimum correlation along the path It corresponds to the regularization parameter alpha parameter in the Lasso

method 'lar' 'lasso' optional default'lar' Specifies the returned model Select lar for Least Angle Regression lasso for the Lasso

copyX bool optional defaultTrue If False Xis overwritten

eps float optional default''npfinfofpfloateps'' The machineprecision regularization in the computation of the Cholesky diagonal factors Increase this for very illconditioned systems

copyGram bool optional defaultTrue If False Gram is overwritten

verbose int default0 Controls output verbosity

returnpath bool optional defaultTrue If returnpathTrue returns the entire path else returns only the last point of the path

returnniter bool optional defaultFalse Whether to return the number of iterations

positive boolean defaultFalse Restrict coefficients to be 0 This option is only allowed with method 'lasso' Note that the model coefficients will not converge to the ordinary leastsquares solution for small values of alpha Only coefficients up to the smallest alpha

1956 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

value `alphas` `alphas` 0min when `fitpath` True reached by the step
wise LarsLasso algorithm are typically in congruence with the solution of the coordinate
descent `lassopath` function

Returns
`alphas` array shape `nalphas` 1 Maximum of covariances in absolute value at each iter
ation `nalphas` is either `maxiter` `nfeatures` or the number of nodes in the path
with `alpha` `alphamin` whichever is smaller
`active` array shape `nalphas` Indices of active variables at the end of the path
`coefs` array shape `nfeatures` `nalphas` 1 Coefficients along the path
`niter` int Number of iterations run Returned only if `returnniter` is set to True

See also
`larspathgram`
`lassopath`
`lassopathgram`
`LassoLars`
`Lars`
`LassoLarsCV`
`LarsCV`
`sklearn.decomposition.sparse_encode`

References
123

Examples using `sklearn.linear_model.lars_path`
• Lasso path using LARS
`62224sklearn.linear_model.lars_pathgram`
`sklearn.linear_model.lars_pathgram` Xy Gram `nsamples` `maxiter` 500 `al`
`phamin` 0 `method` 'lar' `copyX` True
`eps` 2220446049250313e16 `copyGram` True
`verbose` 0 `returnpath` True `returnniter` False
`positive` False
`lars_path` in the sufficient stats mode 1

The optimization objective for the case `method` 'lasso' is
$$\frac{1}{2} \|y - Xw\|_2^2 + \alpha \|w\|_1$$

in the case of `method` 'lars' the objective function is only known in the form of an implicit equation see
discussion in 1
Read more in the User Guide
`622sklearn.linear_model` Generalized Linear Models 1957

scikitlearn user guide Release 0213

Parameters

Xyarraylike shape nsamples or nsamples ntargets Xy npdotXT y
Gram array shape nfeatures nfeatures Gram npdotXT X
nsamples integer or float Equivalent size of sample
maxiter integer optional default500 Maximum number of iterations to perform set to infinity for no limit
alphamin float optional default0 Minimum correlation along the path It corresponds to the regularization parameter alpha parameter in the Lasso
method 'lar' 'lasso' optional default'lar' Specifies the returned model Select lar for Least Angle Regression lasso for the Lasso
copyX bool optional defaultTrue If False Xis overwritten
eps float optional default''npfinfofpfloateps'' The machineprecision regularization in the computation of the Cholesky diagonal factors Increase this for very illconditioned systems
copyGram bool optional defaultTrue If False Gram is overwritten
verbose int default0 Controls output verbosity
returnpath bool optional defaultTrue If returnpathTrue returns the entire path else returns only the last point of the path
returnniter bool optional defaultFalse Whether to return the number of iterations
positive boolean defaultFalse Restrict coefficients to be 0 This option is only allowed with method 'lasso' Note that the model coefficients will not converge to the ordinary leastsquares solution for small values of alpha Only coefficients up to the smallest alpha value alphasalphas 0min when fitpathTrue reached by the step wise LarsLasso algorithm are typically in congruence with the solution of the coordinate descent lassopath function

Returns

alphas array shape nalphas 1 Maximum of covariances in absolute value at each iterationnalphas is eithermaxiter nfeatures or the number of nodes in the path
withalpha alphamin whichever is smaller
active array shape nalphas Indices of active variables at the end of the path
coefs array shape nfeatures nalphas 1 Coefficients along the path
niter int Number of iterations run Returned only if returnniter is set to True

See also

larspath
lassopath
lassopathgram
LassoLars
Lars
LassoLarsCV
LarsCV

scikitlearn user guide Release 0213
sklearn.decomposition.sparse_encode
References
123
62225sklearn.linear_model.LassoPath
sklearn.linear_model.LassoPath Xyeps0001 nalphas100 alphasNone precompute
pute'auto' XyNone copyXTrue coefinitNone verboseFalse returnniterFalse positiveFalse params
Compute Lasso path with coordinate descent
The Lasso optimization function varies for mono and multioutputs
For monooutput tasks it is
1 2 nsamples y Xw22 alpha w1
For multioutput tasks it is
1 2 nsamples Y XW2Fro alpha W21
Where
W21 sum_i sqrt(sum_j w_ij^2)
ie the sum of norm of each row
Read more in the User Guide
Parameters
Xarraylike sparse matrix shape nsamples nfeatures Training data Pass directly as
Fortrancontiguous data to avoid unnecessary memory duplication If yis monooutput then
Xcan be sparse
yndarray shape nsamples or nsamples noutputs Target values
eps float optional Length of the path eps1e3 means thatalphamin alphamax
1e3
nalphas int optional Number of alphas along the regularization path
alphas ndarray optional List of alphas where to compute the models If None alphas are set
automatically
precompute True False 'auto' arraylike Whether to use a precomputed Gram matrix to
speed up calculations If set to auto let us decide The Gram matrix can also be passed
as argument
Xyarraylike optional Xy np.dot(XT y) that can be precomputed It is useful only when
the Gram matrix is precomputed
copyX boolean optional default True If True X will be copied else it may be overwritten
coefinit array shape nfeatures None The initial values of the coefficients
verbose bool or integer Amount of verbosity
returnniter bool whether to return the number of iterations or not
622sklearn.linear_model.GeneralizedLinearModels 1959

scikitlearn user guide Release 0213

positive bool default False If set to True forces coefficients to be positive Only allowed

whenyndim 1

params kwargs keyword arguments passed to the coordinate descent solver

Returns

alphas array shape nalphas The alphas along the path where models are computed

coefs array shape nfeatures nalphas or noutputs nfeatures nalphas Coefficients

along the path

dualgaps array shape nalphas The dual gaps at the end of the optimization for each

alpha

niters arraylike shape nalphas The number of iterations taken by the coordinate descent

optimizer to reach the specified tolerance for each alpha

See also

larspath

Lasso

LassoLars

LassoCV

LassoLarsCV

sklearn.decomposition.sparse_encode

Notes

For an example see `examples/linear_model/plot_lasso_coordinate_descent_path.py`

To avoid unnecessary memory duplication the `X` argument of the fit method should be directly passed as a

Fortran contiguous numpy array

Note that in certain cases the Lars solver may be significantly faster to implement this functionality In particu

lar linear interpolation can be used to retrieve model coefficients between the values output by `larspath`

Examples

Comparing `lassopath` and `larspath` with interpolation

```
X = np.array([1, 2, 3, 1, 2, 3, 5, 4, 4, 3])
```

```
y = np.array([1, 2, 3, 1, 2, 3, 1, 2, 3, 1])
```

Use `lassopath` to compute a coefficient path

```
coef_path = lassopath(X, y, alphas=[5, 1, 5])
```

```
print(coef_path
```

```
0 0 0.46874778
```

```
0.2159048 0.4425765 0.23689075
```

Now use `larspath` and 1D linear interpolation to compute the

same path

```
from sklearn.linear_model import larspath
```

```
alphas, active_coef_path, lars = larspath(X, y, method='lasso')
```

```
from scipy import interpolate
```

```
coef_path_continuous = interpolate.interp1d(alphas, lars)
```

1960 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
coefpathlars 1
printcoefpathcontinuous5 1 5
0 0 046915237
02159048 04425765 023668876
Examples using sklearnlinearmodellassopath
•Lasso and Elastic Net
62226sklearnlinearmodel orthogonalmp
sklearnlinearmodel orthogonalmp XynnonzerocoefsNone tolNone precom
puteFalse copyXTrue returnpathFalse re
turnniterFalse
Orthogonal Matching Pursuit OMP
Solves ntargets Orthogonal Matching Pursuit problems An instance of the problem has the form
When parametrized by the number of nonzero coefficients using nnonzerocoefs argmin y
Xgamma2 subject to gamma0 nnonzero coefs
When parametrized by error using the parameter tol argmin gamma0 subject to y Xgamma2 tol
Read more in the User Guide
Parameters
Xarray shape nsamples nfeatures Input data Columns are assumed to have unit norm
yarray shape nsamples or nsamples ntargets Input targets
nnonzerocoefs int Desired number of nonzero entries in the solution If None by default
this value is set to 10 of nfeatures
tolfloat Maximum norm of the residual If not None overrides nnonzerocoefs
precompute True False ‘auto’ Whether to perform precomputations Improves perfor
mance when ntargets or nsamples is very large
copyX bool optional Whether the design matrix X must be copied by the algorithm A false
value is only helpful if X is already Fortranordered otherwise a copy is made anyway
returnpath bool optional Default False Whether to return every value of the nonzero
coefficients along the forward path Useful for crossvalidation
returnniter bool optional default False Whether or not to return the number of iterations
Returns
coef array shape nfeatures or nfeatures ntargets Coefficients of the OMP solution If
returnpathTrue this contains the whole coefficient path In this case its shape is
nfeatures nfeatures or nfeatures ntargets nfeatures and iterating over the last axis
yields coefficients in increasing order of active features
niters arraylike or int Number of active features across every target Returned only if
returnniter is set to True
See also
OrthogonalMatchingPursuit
622sklearnlinearmodel Generalized Linear Models 1961

scikitlearn user guide Release 0213

orthogonalmpgram

larspath

decompositionsparsencode

Notes

Orthogonal matching pursuit was introduced in S Mallat Z Zhang Matching pursuits with timefrequency dictionaries IEEE Transactions on Signal Processing V ol 41 No 12 December 1993 pp 33973415

<http://blanchepolytechnique.fr/mallat/papiers/MallatPursuit93.pdf>

This implementation is based on Rubinstein R Zibulevsky M and Elad M Efficient Implementation of the KSVD Algorithm using Batch Orthogonal Matching Pursuit Technical Report CS Technion April 2008

<https://www.cs.technion.ac.il/~ronrubin/Publications/KSVDOMPv2.pdf>

62227sklearnlinearmodel orthogonalmpgram

sklearnlinearmodel orthogonalmpgram Gram XynnonzerocoeffsNone tolNone

normssquaredNone copyGramTrue

copyXyTrue returnpathFalse re

turnniterFalse

Gram Orthogonal Matching Pursuit OMP

Solves ntargets Orthogonal Matching Pursuit problems using only the Gram matrix $X^T X$ and the product

$X^T y$

Read more in the User Guide

Parameters

Gram array shape nfeatures nfeatures Gram matrix of the input data $X^T X$

Xyarray shape nfeatures or nfeatures ntargets Input targets multiplied by $X X^T y$

nnonzerocoeffs int Desired number of nonzero entries in the solution If None by default

this value is set to 10 of nfeatures

tolfloat Maximum norm of the residual If not None overrides nnonzerocoeffs

normssquared arraylike shape ntargets Squared L2 norms of the lines of y Required

if tol is not None

copyGram bool optional Whether the gram matrix must be copied by the algorithm A false

value is only helpful if it is already Fortranordered otherwise a copy is made anyway

copyXy bool optional Whether the covariance vector Xy must be copied by the algorithm

If False it may be overwritten

returnpath bool optional Default False Whether to return every value of the nonzero

coefficients along the forward path Useful for crossvalidation

returnniter bool optional default False Whether or not to return the number of iterations

Returns

coef array shape nfeatures or nfeatures ntargets Coefficients of the OMP solution If

returnpathTrue this contains the whole coefficient path In this case its shape is

nfeatures nfeatures or nfeatures ntargets nfeatures and iterating over the last axis

yields coefficients in increasing order of active features

1962 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

nitters arraylike or int Number of active features across every target Returned only if
returnniter is set to True

See also

OrthogonalMatchingPursuit

orthogonalmp

larspath

decompositionsparsencode

Notes

Orthogonal matching pursuit was introduced in G Mallat Z Zhang Matching pursuits with timefrequency
dictionaries IEEE Transactions on Signal Processing V ol 41 No 12 December 1993 pp 33973415

<http://blanchepolytechnique.fr/mallat/papiers/MallatPursuit93.pdf>

This implementation is based on Rubinstein R Zibulevsky M and Elad M Efficient Implementation of

the KSVD Algorithm using Batch Orthogonal Matching Pursuit Technical Report CS Technion April 2008

<https://www.cs.technion.ac.il/~ronrubin/Publications/KSVDOMPv2.pdf>

62228sklearnlinearmodel.ridge.regression

sklearnlinearmodel.ridge.regression.Xyalpha.sampleweight=None solver='auto'

maxiter=None tol=0.0001 verbose=0 random

domstate=None returnniter=False re

turnintercept=False checkinput=True

Solve the ridge equation by the method of normal equations

Read more in the User Guide

Parameters

Xarraylike sparse matrix LinearOperator shape nsamples nfeatures Training data

yarraylike shape nsamples or nsamples ntargets Target values

alpha float arraylike shape ntargets if arraylike Regularization strength must be

a positive float Regularization improves the conditioning of the problem and reduces the

variance of the estimates Larger values specify stronger regularization Alpha corresponds

to C1 in other linear models such as LogisticRegression or LinearSVC If an array is

passed penalties are assumed to be specific to the targets Hence they must correspond in

number

sampleweight float or numpy array of shape nsamples Individual weights for each sam

ple If sampleweight is not None and solver='auto' the solver will be set to 'cholesky'

New in version 0.17

solver 'auto' 'svd' 'cholesky' 'lsqr' 'sparsecg' 'sag' 'saga' Solver to use in the com

putational routines

- 'auto' chooses the solver automatically based on the type of data

- 'svd' uses a Singular Value Decomposition of X to compute the Ridge coefficients More

stable for singular matrices than 'cholesky'

622sklearnlinearmodel.Generalized Linear Models 1963

scikitlearn user guide Release 0213

- ‘cholesky’ uses the standard `scipy.linalg.solve` function to obtain a closedform solution via a Cholesky decomposition of `dot(XT, X)`
 - ‘sparsecg’ uses the conjugate gradient solver as found in `scipysparselinalg.cg`. As an iterative algorithm this solver is more appropriate than ‘cholesky’ for largescale data possibility to set `tol` and `maxiter`
 - ‘lsqr’ uses the dedicated regularized leastsquares routine `scipysparselinalg.lsqr`. It is the fastest and uses an iterative procedure
 - ‘sag’ uses a Stochastic Average Gradient descent and ‘saga’ uses its improved unbiased version named SAGA. Both methods also use an iterative procedure and are often faster than other solvers when both `nsamples` and `nfeatures` are large. Note that ‘sag’ and ‘saga’ fast convergence is only guaranteed on features with approximately the same scale. You can preprocess the data with a scaler from `sklearn.preprocessing`.
- All last five solvers support both dense and sparse data. However only ‘sag’ and ‘sparsecg’ supports sparse input when `fit_intercept` is `True`.

New in version 0.17: Stochastic Average Gradient descent solver

New in version 0.19: SAGA solver

`max_iter` int optional Maximum number of iterations for conjugate gradient solver. For the ‘sparsecg’ and ‘lsqr’ solvers the default value is determined by `scipysparselinalg`. For ‘sag’ and saga solver the default value is 1000.
`tol` float Precision of the solution

`verbose` int Verbosity level. Setting `verbose = 0` will display additional information depending on the solver used.

`random_state` int RandomState instance or None optional default None The seed of the pseudo random number generator to use when shuffling the data. If int `random_state` is the seed used by the random number generator. If RandomState instance `random_state` is the random number generator. If None the random number generator is the RandomState instance used by `nprandom`. Used when solver ‘sag’.

`return_niter` boolean default False If True the method also returns `niter` the actual number of iteration performed by the solver.

New in version 0.17

`return_intercept` boolean default False If True and if `X` is sparse the method also returns the intercept and the solver is automatically changed to ‘sag’. This is only a temporary fix for fitting the intercept with sparse data. For dense data use `sklearn.linear_model.preprocess_data` before your regression.

New in version 0.17

`check_input` boolean default True If False the input arrays `X` and `y` will not be checked.

New in version 0.21

Returns
`coef` array shape `nfeatures` or `ntargets` `nfeatures` Weight vectors
`niter` int optional The actual number of iteration performed by the solver. Only returned if `return_niter` is True

`intercept` float or array shape `ntargets` The intercept of the model. Only returned if `return_intercept` is True and if `X` is a scipy sparse array

1964 Chapter 6: API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Notes

This function won't compute the intercept

623sklearnmanifold Manifold Learning

Thesklearnmanifold module implements data embedding techniques

User guide See the Manifold learning section for further details

manifoldIsomap nneighbors ncomponents Isomap Embedding

manifoldLocallyLinearEmbedding Locally Linear Embedding

manifoldMDS ncomponents metric ninit Multidimensional scaling

manifoldSpectralEmbedding ncomponents

Spectral embedding for nonlinear dimensionality reduction

manifoldTSNE ncomponents perplexity tdistributed Stochastic Neighbor Embedding

6231sklearnmanifold Isomap

classsklearnmanifold Isomapnneighbors5 ncomponents2 eigensolver'auto' tol0

maxiterNone pathmethod'auto' neighborsalgorithm'auto'

njobsNone

Isomap Embedding

Nonlinear dimensionality reduction through Isometric Mapping

Read more in the User Guide

Parameters

nneighbors integer number of neighbors to consider for each point

ncomponents integer number of coordinates for the manifold

eigensolver 'auto' 'arnpack' 'dense' 'auto' Attempt to choose the most efficient solver for the given problem

'arnpack' Use Arnoldi decomposition to find the eigenvalues and eigenvectors

'dense' Use a direct solver ie LAPACK for the eigenvalue decomposition

tolfloat Convergence tolerance passed to arpack or lobpcg not used if eigensolver

'dense'

maxiter integer Maximum number of iterations for the arpack solver not used if

eigensolver 'dense'

pathmethod string 'auto' 'FW' 'D' Method to use in finding shortest path

'auto' attempt to choose the best algorithm automatically

'FW' FloydWarshall algorithm

'D' Dijkstra's algorithm

neighborsalgorithm string 'auto' 'brute' 'kdtree' 'balltree' Algorithm to use for nearest

neighbors search passed to neighborsNearestNeighbors instance

623sklearnmanifold Manifold Learning 1965

scikitlearn user guide Release 0213

njobs int or None optional defaultNone The number of parallel jobs to run None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

Attributes

embedding arraylike shape nsamples ncomponents Stores the embedding vectors

kernelpca objectKernelPCA object used to implement the embedding

trainingdata arraylike shape nsamples nfeatures Stores the training data

nbrs sklearnneighborsNearestNeighbors instance Stores nearest neighbors instance including BallTree or KDtree if applicable

dismatrix arraylike shape nsamples nsamples Stores the geodesic distance matrix of training data

References

R7f4d308f50541

Examples

```
from sklearn.datasets import load_digits
from sklearn.manifold import Isomap
X, load_digitsreturnXy = True
Xshape
1797 64
embedding Isomapncomponents2
Xtransformed embeddingfittransformX100
Xtransformedshape
100 2
```

Methods

fitself X y Compute the embedding vectors for data X

fittransform self X y Fit the model from data in X and transform X

getparams self deep Get parameters for this estimator

reconstructionerror self Compute the reconstruction error for the embedding

setparams self params Set the parameters of this estimator

transform self X Transform X

init selfnneighbors5 ncomponents2 eigensolver'auto' tol0 maxiterNone pathmethod'auto' neighborsalgorithm'auto' njobsNone

fitselfXyNone

Compute the embedding vectors for data X

Parameters

Xarraylike sparse matrix BallTree KDTree NearestNeighbors Sample data shape nsamples nfeatures in the form of a numpy array precomputed tree or Nearest Neighbors object

1966 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

ylgnored

Returns

self returns an instance of self

fittransform selfXyNone

Fit the model from data in X and transform X

Parameters

Xarraylike sparse matrix BallTree KDTree Training vector where nsamples in the number of samples and nfeatures is the number of features

ylgnored

Returns

Xnew arraylike shape nsamples ncomponents

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

reconstructionerror self

Compute the reconstruction error for the embedding

Returns

reconstructionerror float

Notes

The cost function of an isomap embedding is

$E = \text{frobeniusnorm}(KD - KD_{\text{fit}})$ / nsamples

Where D is the matrix of distances for the input data X Dfit is the matrix of distances for the output

embedding Xfit and K is the isomap kernel

$KD = \frac{1}{n_1 n_2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} D_{ij}^2$ / nsamples

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Transform X

623sklearnmanifold Manifold Learning 1967

scikitlearn user guide Release 0213

This is implemented by linking the points X into the graph of geodesic distances of the training data. First then nearest neighbors of X are found in the training data and from these the shortest geodesic distances from each point in X to each point in the training data are computed in order to construct the kernel. The embedding of X is the projection of this kernel onto the embedding vectors of the training set.

Parameters

X arraylike shape nsamples nfeatures

Returns

Xnew arraylike shape nsamples ncomponents

Examples using sklearnmanifoldIsomap

- Comparison of Manifold Learning methods
- Manifold Learning methods on a severed sphere
- Manifold learning on handwritten digits Locally Linear Embedding Isomap

6232sklearnmanifold LocallyLinearEmbedding
class sklearnmanifold LocallyLinearEmbedding nneighbors5 ncomponents2

reg0001 eigensolver'auto' tol1e

06maxiter100 method'standard'

hessiantol00001 modifiedtol1e

12 neighborsalgorithm'auto' ran

domstateNone njobsNone

Locally Linear Embedding

Read more in the User Guide

Parameters

nneighbors integer number of neighbors to consider for each point

ncomponents integer number of coordinates for the manifold

regfloat regularization constant multiplies the trace of the local covariance matrix of the distances

eigensolver string 'auto' 'arpack' 'dense' auto algorithm will attempt to choose the best method for input data

arpack use arnoldi iteration in shiftinvert mode For this method M may be a dense matrix sparse matrix or general linear operator Warning ARPACK can be unstable for some problems It is best to try several random seeds in order to check results

dense use standard dense matrix operations for the eigenvalue decomposition For this method M must be an array or matrix type This method should be avoided for large problems

tolfloat optional Tolerance for 'arpack' method Not used if eigensolver'dense'

maxiter integer maximum number of iterations for the arpack solver Not used if eigensolver'dense'

method string 'standard' 'hessian' 'modified' or 'ltsa'

standard use the standard locally linear embedding algorithm see reference 1

1968 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

hessian use the Hessian eigenmap method This method requires nneighbors

ncomponents 1 ncomponents 1 2 see reference 2

modified use the modified locally linear embedding algorithm see reference 3

ltsa use local tangent space alignment algorithm see reference 4

hessian_tol float optional Tolerance for Hessian eigenmapping method Only used if

method hessian

modified_tol float optional Tolerance for modified LLE method Only used if method

modified

neighbors_algorithm string 'auto''brute''kdtree''balltree' algorithm to use for nearest

neighbors search passed to neighborsNearestNeighbors instance

random_state int RandomState instance or None optional defaultNone If int ran

dom_state is the seed used by the random number generator If RandomState instance ran

dom_state is the random number generator If None the random number generator is the

RandomState instance used by np.random Used when eigensolver 'arpack'

njobs int or None optional defaultNone The number of parallel jobs to run None means

1 unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

Attributes

embedding arraylike shape nsamples ncomponents Stores the embedding vectors

reconstruction_error float Reconstruction error associated with embedding

nbrs NearestNeighbors object Stores nearest neighbors instance including BallTree or

KDtree if applicable

References

R62e36dd1b0561 R62e36dd1b0562 R62e36dd1b0563 R62e36dd1b0564

Examples

from sklearn.datasets import load_digits

from sklearn.manifold import LocallyLinearEmbedding

X, load_digits.return_Xy True

Xshape

1797 64

embedding LocallyLinearEmbedding.ncomponents2

Xtransformed embedding.fit_transform(X)100

Xtransformedshape

100 2

Methods

fitself X y Compute the embedding vectors for data X

fit_transform self X y Compute the embedding vectors for data X and trans

form X

Continued on next page

623sklearn.manifold Manifold Learning 1969

getparams self deep Get parameters for this estimator
setparams self params Set the parameters of this estimator
transform self X Transform new points into embedding space
init selfnneighbors5 ncomponents2 reg0001 eigensolver'auto' tol1e06
maxiter100 method'standard' hessiantol00001 modifiedtol1e12 neigh
borsalgorithm'auto' randomstateNone njobsNone
fitselfXyNone

Compute the embedding vectors for data X

Parameters

Xarraylike of shape nsamples nfeatures training set

yIgnored

Returns

self returns an instance of self

fittransform selfXyNone

Compute the embedding vectors for data X and transform X

Parameters

Xarraylike of shape nsamples nfeatures training set

yIgnored

Returns

Xnew arraylike shape nsamples ncomponents

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Transform new points into embedding space

Parameters

Xarraylike shape nsamples nfeatures

Returns

scikitlearn user guide Release 0213

Xnew array shape nsamples ncomponents

Notes

Because of scaling performed by this method it is discouraged to use it together with methods that are not scaleinvariant like SVMs

Examples using sklearnmanifoldLocallyLinearEmbedding

- Visualizing the stock market structure
- Comparison of Manifold Learning methods
- Manifold Learning methods on a severed sphere
- Manifold learning on handwritten digits Locally Linear Embedding Isomap

6233sklearnmanifold MDS

classssklearnmanifold MDSncomponents2 metricTrue ninit4 maxiter300 verbose0 eps0001 njobsNone randomstateNone dissimilar

ity'euclidean'

Multidimensional scaling

Read more in the User Guide

Parameters

ncomponents int optional default 2 Number of dimensions in which to immerse the dis similarities

metric boolean optional default True If True perform metric MDS otherwise perform nonmetric MDS

ninit int optional default 4 Number of times the SMACOF algorithm will be run with different initializations The final results will be the best output of the runs determined by the run with the smallest final stress

maxiter int optional default 300 Maximum number of iterations of the SMACOF algo rithm for a single run

verbose int optional default 0 Level of verbosity

eps float optional default 1e3 Relative tolerance with respect to stress at which to declare convergence

njobs int or None optional defaultNone The number of jobs to use for the computation If multiple initializations are used ninit each run of the algorithm is computed in parallel

None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

randomstate int RandomState instance or None optional default None The generator used to initialize the centers If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

6233sklearnmanifold Manifold Learning 1971

scikitlearn user guide Release 0213

dissimilarity ‘euclidean’ ‘precomputed’ optional default ‘euclidean’ Dissimilarity measure to use

- ‘euclidean’ Pairwise Euclidean distances between points in the dataset
- ‘precomputed’ Precomputed dissimilarities are passed directly to fit and fittransform

Attributes

embedding arraylike shape nsamples ncomponents Stores the position of the dataset in the embedding space

stress float The final value of the stress sum of squared distance of the disparities and the distances for all constrained points

References

“Modern Multidimensional Scaling Theory and Applications” Borg I Groenen P Springer Series in Statistics 1997

“Nonmetric multidimensional scaling a numerical method” Kruskal J Psychometrika 29 1964

“Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis” Kruskal J Psychometrika 29 1964

Examples

```
from sklearn.datasets import load_digits
from sklearn.manifold import MDS
X = load_digits().X
Xshape = X.shape
1797 64
embedding = MDS(n_components=2)
X_transformed = embedding.fit_transform(X)
X_transformedshape = X_transformed.shape
100 2
```

Methods

fitself X y init Computes the position of the points in the embedding space

fittransform self X y init Fit the data from X and returns the embedded coordinates

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

init self n_components=2 metric=True n_init=4 max_iter=300 verbose=0 eps=0.001 n_jobs=None random_state=None dissimilarity='euclidean'

fitself X y None init None

Computes the position of the points in the embedding space

Parameters

1972 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarray shape nsamples nfeatures or nsamples nsamples Input data If
dissimilarityprecomputed the input should be the dissimilarity matrix
yIgnored

init ndarray shape nsamples optional default None Starting configuration of the em
bedding to initialize the SMACOF algorithm By default the algorithm is initialized with
a randomly chosen array
fittransform selfXyNone initNone

Fit the data from X and returns the embedded coordinates

Parameters

Xarray shape nsamples nfeatures or nsamples nsamples Input data If
dissimilarityprecomputed the input should be the dissimilarity matrix
yIgnored

init ndarray shape nsamples optional default None Starting configuration of the em
bedding to initialize the SMACOF algorithm By default the algorithm is initialized with
a randomly chosen array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it's possible to update each component
of a nested object

Returns

self

Examples using sklearnmanifoldMDS

- Multidimensional scaling
 - Comparison of Manifold Learning methods
 - Manifold Learning methods on a severed sphere
 - Manifold learning on handwritten digits Locally Linear Embedding Isomap
- 623sklearnmanifold Manifold Learning 1973

scikitlearn user guide Release 0213

6234sklearnmanifold SpectralEmbedding

classsklearnmanifold SpectralEmbedding ncomponents2 affinity'nearestneighbors'

gammaNone randomstateNone

eigensolverNone nneighborsNone

njobsNone

Spectral embedding for nonlinear dimensionality reduction

Forms an affinity matrix given by the specified function and applies spectral decomposition to the corresponding

graph laplacian The resulting transformation is given by the value of the eigenvectors for each data point

Note Laplacian Eigenmaps is the actual algorithm implemented here

Read more in the User Guide

Parameters

ncomponents integer default 2 The dimension of the projected subspace

affinity string or callable default

How to construct the affinity matrix

- 'nearestneighbors' construct affinity matrix by knn graph
- 'rbf' construct affinity matrix by rbf kernel
- 'precomputed' interpret X as precomputed affinity matrix
- callable use passed in function as affinity the function takes in data matrix nsamples

nfeatures and return affinity matrix nsamples nsamples

gamma float optional default Kernel coefficient for rbf kernel

randomstate int RandomState instance or None optional default None A pseudo random

number generator used for the initialization of the lobpcg eigenvectors If int randomstate is

the seed used by the random number generator If RandomState instance randomstate is

the random number generator If None the random number generator is the RandomState

instance used by nprandom Used when solver 'amg'

eigensolver None 'arpack' 'lobpcg' or 'amg' The eigenvalue decomposition strategy to

use AMG requires pyamg to be installed It can be faster on very large sparse problems

but may also lead to instabilities

nneighbors int default Number of nearest neighbors for nearestneighbors graph building

njobs int or None optional defaultNone The number of parallel jobs to run None means

1 unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

Attributes

embedding array shape nsamples ncomponents Spectral embedding of the training

matrix

affinitymatrix array shape nsamples nsamples Affinitymatrix constructed from

samples or precomputed

References

- A Tutorial on Spectral Clustering 2007 Ulrike von Luxburg <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.165.9323>

1974 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- On Spectral Clustering Analysis and an algorithm 2001 Andrew Y Ng Michael I Jordan Yair Weiss <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.198.100>
- Normalized cuts and image segmentation 2000 Jianbo Shi Jitendra Malik <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.160.2324>

Examples

```
from sklearn.datasets import load_digits
from sklearn.manifold import SpectralEmbedding
X, load_digits.return_Xy = True
```

```
X.shape
(1797, 64)
embedding = SpectralEmbedding(n_components=2)
X_transformed = embedding.fit_transform(X[100])
X_transformed.shape
(100, 2)
```

Methods

```
fit(self, X, y) Fit the model from data in X
fit_transform(self, X, y) Fit the model from data in X and transform X
get_params(self, deep=True) Get parameters for this estimator
set_params(self, **params) Set the parameters of this estimator
init(self, n_components=2, affinity='nearest_neighbors', gamma=None, random_state=None)
eigen_solver='none', n_neighbors=None, n_jobs=None)
fit(self, X, y=None)
```

Fit the model from data in X

Parameters

X: array-like shape (n_samples, n_features) Training vector where n_samples is the number of samples and n_features is the number of features

If affinity is "precomputed" X: array-like shape (n_samples, n_samples) Interpret X as precomputed adjacency graph computed from samples

Returns

```
self object Returns the instance itself
fit_transform(self, X, y=None)
```

Fit the model from data in X and transform X

Parameters

X: array-like shape (n_samples, n_features) Training vector where n_samples is the number of samples and n_features is the number of features

If affinity is "precomputed" X: array-like shape (n_samples, n_samples) Interpret X as precomputed adjacency graph computed from samples

Returns

```
X_new: array-like shape (n_samples, n_components)
(623, 2) sklearn.manifold.Manifold Learning 1975
```

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnmanifoldSpectralEmbedding

- Various Agglomerative Clustering on a 2D embedding of digits

- Comparison of Manifold Learning methods

- Manifold Learning methods on a severed sphere

- Manifold learning on handwritten digits Locally Linear Embedding Isomap

6235sklearnmanifold TSNE

classssklearnmanifold TSNEcomponents2 perplexity300 earlyexaggeration120

learningrate2000 niter1000 niterwithoutprogress300

mingradnorm1e07 metric'euclidean' init'random' verbose0

randomstateNone method'barneshut' angle05

tdistributed Stochastic Neighbor Embedding

tSNE 1 is a tool to visualize highdimensional data It converts similarities between data points to joint

probabilities and tries to minimize the KullbackLeibler divergence between the joint probabilities of the low

dimensional embedding and the highdimensional data tSNE has a cost function that is not convex ie with

different initializations we can get different results

It is highly recommended to use another dimensionality reduction method eg PCA (Principal Component Analysis) for dense data or Truncat

edSVD for sparse data to reduce the number of dimensions to a reasonable amount eg 50 if the number of

features is very high This will suppress some noise and speed up the computation of pairwise distances between

samples For more tips see Laurens van der Maaten's FAQ 2

Read more in the User Guide

Parameters

ncomponents int optional default 2 Dimension of the embedded space

perplexity float optional default 30 The perplexity is related to the number of nearest

neighbors that is used in other manifold learning algorithms Larger datasets usually require

a larger perplexity Consider selecting a value between 5 and 50 Different values can result

in significantly different results

1976 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

earlyexaggeration float optional default 120 Controls how tight natural clusters in the original space are in the embedded space and how much space will be between them For larger values the space between natural clusters will be larger in the embedded space Again the choice of this parameter is not very critical If the cost function increases during initial optimization the early exaggeration factor or the learning rate might be too high learningrate float optional default 2000 The learning rate for tSNE is usually in the range 100 10000 If the learning rate is too high the data may look like a 'ball' with any point approximately equidistant from its nearest neighbours If the learning rate is too low most points may look compressed in a dense cloud with few outliers If the cost function gets stuck in a bad local minimum increasing the learning rate may help niter int optional default 1000 Maximum number of iterations for the optimization Should be at least 250

niterwithoutprogress int optional default 300 Maximum number of iterations without progress before we abort the optimization used after 250 initial iterations with early exaggeration Note that progress is only checked every 50 iterations so this value is rounded to the next multiple of 50

New in version 017 parameter niterwithoutprogress to control stopping criteria mingradnorm float optional default 1e7 If the gradient norm is below this threshold the optimization will be stopped

metric string or callable optional The metric to use when calculating distance between instances in a feature array If metric is a string it must be one of the options allowed by scipyspatialdistancepdist for its metric parameter or a metric listed in pairwisePAIRWISEDISTANCEFUNCTIONS If metric is "precomputed" X is assumed to be a distance matrix Alternatively if metric is a callable function it is called on each pair of instances rows and the resulting value recorded The callable should take two arrays from X as input and return a value indicating the distance between them The default is "euclidean" which is interpreted as squared euclidean distance

init string or numpy array optional default "random" Initialization of embedding Possible options are 'random' 'pca' and a numpy array of shape nsamples ncomponents PCA (Principal Component Analysis) initialization cannot be used with precomputed distances and is usually more globally stable than random initialization

verbose int optional default 0 Verbosity level

randomstate int RandomState instance or None optional default None If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom Note that different initializations might result in different local minima of the cost function

method string default 'barneshut' By default the gradient calculation algorithm uses BarnesHut approximation running in $O(N \log N)$ time method'exact' will run on the slower but exact algorithm in $O(N^2)$ time The exact algorithm should be used when nearestneighbor errors need to be better than 3 However the exact method cannot scale to millions of examples

New in version 017 Approximate optimization method via the BarnesHut

angle float default 05 Only used if method'barneshut' This is the tradeoff between speed and accuracy for BarnesHut TSNE 'angle' is the angular size referred to as theta in 3 of a distant node as measured from a point If this size is below 'angle' then it is used as a summary node of all points contained within it This method is not very sensitive to 623sklearnmanifold Manifold Learning 1977

scikitlearn user guide Release 0213

changes in this parameter in the range of 0.2 - 0.8 Angle less than 0.2 has quickly increasing computation time and angle greater 0.8 has quickly increasing error

Attributes

embedding arraylike shape n_samples n_components Stores the embedding vectors

kldivergence float KullbackLeibler divergence after optimization

niter int Number of iterations run

References

1 van der Maaten LJP Hinton GE Visualizing HighDimensional Data Using tSNE Journal of Machine Learning Research 925792605 2008

2 van der Maaten LJP tDistributed Stochastic Neighbor Embedding <https://lvdmaaten.github.io/tsne>

3 LJP van der Maaten Accelerating tSNE using TreeBased Algorithms Journal of Machine Learning Research 15Oct32213245 2014 <https://lvdmaaten.github.io/publications/papers/JMLR2014.pdf>

Examples

```
import numpy as np
from sklearn.manifold import TSNE
X = np.array([0, 0, 0, 1, 1, 1, 0, 1, 1, 1])
X_embedded = TSNE(n_components=2).fit_transform(X)
X_embedded.shape
```

4 2

Methods

fitself X y Fit X into an embedded space

fittransform self X y Fit X into an embedded space and return that transformed output

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

init self n_components=2 perplexity=300 early_exaggeration=120 learning_rate=2000

niter=1000 n_iter_without_progress=300 min_grad_norm=1e-07 metric='euclidean'

init='random' verbose=0 random_state=None method='barnes_hut' angle=0.5

fitself X y None

Fit X into an embedded space

Parameters

X array shape n_samples n_features or n_samples n_samples If the metric is 'precomputed' X must be a square distance matrix Otherwise it contains a sample per row If the method is 'exact' X may be a sparse matrix of type 'csr' 'csc' or 'coo'

ylgnored

fittransform self X y None

Fit X into an embedded space and return that transformed output

1978 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xarray shape nsamples nfeatures or nsamples nsamples If the metric is 'pre computed' X must be a square distance matrix Otherwise it contains a sample per row

ylgnored

Returns

Xnew array shape nsamples ncomponents Embedding of the training data in low dimensional space

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnmanifoldTSNE

- tSNE The effect of various perplexity values on the shape
- Comparison of Manifold Learning methods
- Manifold Learning methods on a severed sphere
- Manifold learning on handwritten digits Locally Linear Embedding Isomap

manifoldlocallylinearembdding X

Perform a Locally Linear Embedding analysis on the data

manifoldsmacof dissimilarities metric Computes multidimensional scaling using the SMACOF algorithm

manifoldspectralembedding adjacency Project the sample on the first eigenvectors of the graph

Laplacian

6236sklearnmanifold locallylinearembdding

sklearnmanifold locallylinearembdding Xnneighbors ncomponents reg0001

eigensolver'auto' tol1e06 maxiter100

method'standard' hessiantol00001

modifiedtol1e12 randomstateNone

njobsNone

Perform a Locally Linear Embedding analysis on the data

623sklearnmanifold Manifold Learning 1979

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

Xarraylike NearestNeighbors Sample data shape nsamples nfeatures in the form of a numpy array or a NearestNeighbors object

nneighbors integer number of neighbors to consider for each point

ncomponents integer number of coordinates for the manifold

regfloat regularization constant multiplies the trace of the local covariance matrix of the distances

eigensolver string ‘auto’ ‘arpack’ ‘dense’ auto algorithm will attempt to choose the best method for input data

arpack use arnoldi iteration in shiftinvert mode For this method M may be a dense matrix sparse matrix or general linear operator Warning ARPACK can be unstable for some problems It is best to try several random seeds in order to check results

dense use standard dense matrix operations for the eigenvalue decomposition For this method M must be an array or matrix type This method should be avoided for large problems

tolfloat optional Tolerance for ‘arpack’ method Not used if eigensolver ‘dense’

maxiter integer maximum number of iterations for the arpack solver

method ‘standard’ ‘hessian’ ‘modified’ ‘ltsa’

standard use the standard locally linear embedding algorithm see reference 1

hessian use the Hessian eigenmap method This method requires nneighbors

ncomponents 1 ncomponents 1 2 see reference 2

modified use the modified locally linear embedding algorithm see reference 3

ltsa use local tangent space alignment algorithm see reference 4

hessian_tol float optional Tolerance for Hessian eigenmapping method Only used if method ‘hessian’

modified_tol float optional Tolerance for modified LLE method Only used if method ‘modified’

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random Used when solver ‘arpack’

njobs int or None optional defaultNone The number of parallel jobs to run for neighbors searchNone means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

Returns

Yarraylike shape nsamples ncomponents Embedding vectors

squarederror float Reconstruction error for the embedding vectors Equivalent to normY - W Y fro 2 where W are the reconstruction weights

1980 Chapter 6 API (Application Programming Interface) Reference

Examples using sklearnmanifoldlocallylinearembdding

•Swiss Roll reduction with LLE

6237sklearnmanifold smacof

sklearnmanifold smacofdissimilarities metricTrue ncomponents2 initNone ninit8

njobsNone maxiter300 verbose0 eps0001 randomstateNone

returnniterFalse

Computes multidimensional scaling using the SMACOF algorithm

The SMACOF Scaling by MAjorizing a COmplicated Function algorithm is a multidimensional scaling algo
rithm which minimizes an objective function the stress using a majorization technique Stress majorization
also known as the Guttman Transform guarantees a monotone convergence of stress and is more powerful than
traditional techniques such as gradient descent

The SMACOF algorithm for metric MDS can summarized by the following steps

1 Set an initial start configuration randomly or not

2 Compute the stress

3 Compute the Guttman Transform

4 Iterate 2 and 3 until convergence

The nonmetric algorithm adds a monotonic regression step before computing the stress

Parameters

dissimilarities ndarray shape nsamples nsamples Pairwise dissimilarities between the
points Must be symmetric

metric boolean optional default True Compute metric or nonmetric SMACOF algorithm
ncomponents int optional default 2 Number of dimensions in which to immerse the dis
similarities If an init array is provided this option is overridden and the shape of init
is used to determine the dimensionality of the embedding space

init ndarray shape nsamples ncomponents optional default None Starting configura
tion of the embedding to initialize the algorithm By default the algorithm is initialized
with a randomly chosen array

ninit int optional default 8 Number of times the SMACOF algorithm will be run with
different initializations The final results will be the best output of the runs determined by
the run with the smallest final stress If init is provided this option is overridden and a
single run is performed

njobs int or None optional defaultNone The number of jobs to use for the computation
If multiple initializations are used ninit each run of the algorithm is computed in
parallel

None means 1 unless in a joblibparallelbackend context1means using all
processors See Glossary for more details

623sklearnmanifold Manifold Learning 1981

scikitlearn user guide Release 0213

maxiter int optional default 300 Maximum number of iterations of the SMACOF algorithm for a single run

verbose int optional default 0 Level of verbosity

eps float optional default 1e3 Relative tolerance with respect to stress at which to declare convergence

randomstate int RandomState instance or None optional default None The generator used to initialize the centers If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by np.random

return_niter bool optional default False Whether or not to return the number of iterations

Returns
X ndarray shape (n_samples, n_components) Coordinates of the points in a n_components space
stress float The final value of the stress sum of squared distance of the disparities and the distances for all constrained points
niter int The number of iterations corresponding to the best stress Returned only if return_niter is set to True

Notes
“Modern Multidimensional Scaling Theory and Applications” Borg I Groenen P Springer Series in Statistics 1997

“Nonmetric multidimensional scaling a numerical method” Kruskal J Psychometrika 29 1964

“Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis” Kruskal J Psychometrika 29 1964

6238 sklearn.manifold.SpectralEmbedding
sklearn.manifold.SpectralEmbedding adjacency n_components=8 eigensolver=None random_state=None epsilon=1e-05 norm_laplacian=True drop_first=True

Project the sample on the first eigenvectors of the graph Laplacian
The adjacency matrix is used to compute a normalized graph Laplacian whose spectrum especially the eigen vectors associated to the smallest eigenvalues has an interpretation in terms of minimal number of cuts necessary to split the graph into comparably sized components
This embedding can also ‘work’ even if the adjacency variable is not strictly the adjacency matrix of a graph but more generally an affinity or similarity matrix between samples for instance the heat kernel of a euclidean distance matrix or a kNN matrix
However care must be taken to always make the affinity matrix symmetric so that the eigenvector decomposition works as expected

Note Laplacian Eigenmaps is the actual algorithm implemented here
Read more in the User Guide

Parameters
1982 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

adjacency arraylike or sparse matrix shape nsamples nsamples The adjacency matrix of the graph to embed

ncomponents integer optional default 8 The dimension of the projection subspace

eigensolver None ‘arpack’ ‘lobpcg’ or ‘amg’ default None The eigenvalue decomp

sition strategy to use AMG requires pyamg to be installed It can be faster on very large

sparse problems but may also lead to instabilities

randomstate int RandomState instance or None optional default None A pseudo random

number generator used for the initialization of the lobpcg eigenvectors decomposition If int

randomstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the

RandomState instance used by nprandom Used when solver ‘amg’

eigentol float optional default00 Stopping criterion for eigendecomposition of the Lapla
cian matrix when using arpack eigensolver

normlaplacian bool optional defaultTrue If True then compute normalized Laplacian

dropfirst bool optional defaultTrue Whether to drop the first eigenvector For spectral em

bedding this should be True as the first eigenvector should be constant vector for connected

graph but for spectral clustering this should be kept as False to retain the first eigenvector

Returns

embedding array shapensamples ncomponents The reduced samples

Notes

Spectral Embedding Laplacian Eigenmaps is most useful when the graph has one connected component If

there graph has many components the first few eigenvectors will simply uncover the connected components of
the graph

References

• <https://en.wikipedia.org/wiki/LOBPCG>

• Toward the Optimal Preconditioned Eigensolver Locally Optimal Block Preconditioned Conjugate Gra
dient Method Andrew V Knyazev <https://doi.org/10.1137/2FS1064827500366124>

624sklearnmetrics Metrics

See the Model evaluation quantifying the quality of predictions section and the Pairwise metrics Affinities and

Kernels section of the user guide for further details The sklearnmetrics module includes score functions

performance metrics and pairwise metrics and distance computations

6241 Model Selection Interface

See the The scoring parameter defining model evaluation rules section of the user guide for further details

metricscheckscoring estimator scoring Determine scorer from user options

metricsgetscorer scoring Get a scorer from string

Continued on next page

624sklearnmetrics Metrics 1983

metricsmakescorer scorefunc Make a scorer from a performance metric or loss function

sklearnmetrics checkscoring

sklearnmetrics checkscoring estimator scoringNone allownoneFalse

Determine scorer from user options

A TypeError will be thrown if the estimator cannot be scored

Parameters

estimator estimator object implementing ‘fit’ The object to use to fit the data

scoring string callable or None optional default None A string see model evaluation doc
umentation or a scorer callable object function with signature scorerestimator

X y

allownone boolean optional default False If no scoring is specified and the estimator has
no score function we can either return None or raise an exception

Returns

scoring callable A scorer callable object function with signature scorerestimator

X y

sklearnmetrics getscorer

sklearnmetrics getscorer scoring

Get a scorer from string

Parameters

scoring str callable scoring method as string If callable it is returned as is

Returns

scorer callable The scorer

sklearnmetrics makescorer

sklearnmetrics makescorer scorefunc greaterisbetterTrue needsprobaFalse

needsthresholdFalse kwargs

Make a scorer from a performance metric or loss function

This factory function wraps scoring functions for use in GridSearchCV and crossvalscore It takes

a score function such as accuracyscore meansquarederror adjustedrandindex or

averageprecision and returns a callable that scores an estimator’s output

Read more in the User Guide

Parameters

scorefunc callable Score function or loss function with signature scorefuncy

ypred kwargs

greaterisbetter boolean defaultTrue Whether scorefunc is a score function default

meaning high is good or a loss function meaning low is good In the latter case the scorer

object will signflip the outcome of the scorefunc

scikitlearn user guide Release 0213

needsproba boolean defaultFalse Whether scorefunc requires predictproba to get probability estimates out of a classifier

If True for binary ytrue the score function is supposed to accept a 1D ypred ie probability of the positive class shape nsamples

needsthreshold boolean defaultFalse Whether scorefunc takes a continuous decision certainty This only works for binary classification using estimators that have either a decisionfunction or predictproba method

If True for binary ytrue the score function is supposed to accept a 1D ypred ie probability of the positive class or the decision function shape nsamples

For example averageprecision or the area under the roc curve can not be computed using discrete predictions alone

kwargs additional arguments Additional parameters to be passed to scorefunc

Returns

scorer callable Callable object that returns a scalar score greater is better

Examples

```
from sklearn.metrics import fbeta_score, make_scorer
ftwo_scorer = make_scorer(fbeta_score, beta=2)
ftwo_scorer
make_scorer(fbeta_score, beta=2)
from sklearn.model_selection import GridSearchCV
from sklearn.svm import LinearSVC
grid = GridSearchCV(LinearSVC(), param_grid={'C': 1, 'gamma': 10},
                    scoring=ftwo_scorer)
```

Examples using sklearn.metrics.make_scorer

- Demonstration of multi-metric evaluation on cross_val_score and GridSearchCV

6242 Classification metrics

See the Classification metrics section of the user guide for further details

metrics.accuracy_score(y_true, y_pred) Accuracy classification score

metrics.auc(x, y) Compute Area Under the Curve (AUC) using the trapezoidal rule

metrics.average_precision_score(y_true, y_score) Compute average precision (AP) from prediction scores

metrics.balanced_accuracy_score(y_true, y_pred) Compute the balanced accuracy

metrics.brier_score_loss(y_true, y_prob) Compute the Brier score

metrics.classification_report(y_true, y_pred) Build a text report showing the main classification metrics

metrics.cohen_kappa_score(y1, y2, labels) Cohen's kappa a statistic that measures inter-annotator agreement

Continued on next page

624sklearn.metrics Metrics 1985

`metricsconfusionmatrix ytrue ypred` Compute confusion matrix to evaluate the accuracy of a classification

`metricsf1score ytrue ypred labels` Compute the F1 score also known as balanced Fscore or Fmeasure

`metricsfbetascore ytrue ypred beta` Compute the Fbeta score

`metricshammingloss ytrue ypred` Compute the average Hamming loss

`metricshingeloss ytrue preddecision` Average hinge loss nonregularized

`metricsjaccardscore ytrue ypred` Jaccard similarity coefficient score

`metricslogloss ytrue ypred eps` Log loss aka logistic loss or crossentropy loss

`metricsmatthewscorrcoef ytrue ypred`

 Compute the Matthews correlation coefficient MCC

`metricsmultilabelconfusionmatrix ytrue`

 Compute a confusion matrix for each class or sample

`metricsprecisionrecallcurve ytrue` Compute precisionrecall pairs for different probability thresholds

`metricsprecisionrecallfscoresupport` Compute precision recall Fmeasure and support for each class

`metricsprecisionscore ytrue ypred` Compute the precision

`metricsrecallscore ytrue ypred` Compute the recall

`metricsrocaucscore ytrue yscore` Compute Area Under the Receiver Operating Characteristic Curve ROC AUC from prediction scores

`metricsroccurve ytrue yscore` Compute Receiver operating characteristic ROC

`metricszerooneloss ytrue ypred` Zeroone classification loss

`sklearnmetrics accuracyscore`

`sklearnmetrics accuracyscore ytrue ypred normalizeTrue sampleweightNone`

 Accuracy classification score

In multilabel classification this function computes subset accuracy the set of labels predicted for a sample must exactly match the corresponding set of labels in ytrue

Read more in the User Guide

Parameters

`ytrue` 1d arraylike or label indicator array sparse matrix Ground truth correct labels

`ytrue` 1d arraylike or label indicator array sparse matrix Predicted labels as returned by a classifier

`normalize` bool optional defaultTrue If False return the number of correctly classified

`samples` Otherwise return the fraction of correctly classified samples

`sampleweight` arraylike of shape nsamples optional Sample weights

Returns

`score` float If`normalize` True return the fraction of correctly classified samples

float else returns the number of correctly classified samples int

The best performance is 1 with `normalize` True and the number of samples with

`normalize` False

See also

`jaccardscore` `hammingloss` `zerooneloss`

scikitlearn user guide Release 0213

Notes

In binary and multiclass classification this function is equal to the jaccardscore function

Examples

```
from sklearn.metrics import accuracyscore
ypred 0 2 1 3
ytrue 0 1 2 3
accuracyscoreytrue ypred
05
accuracyscoreytrue ypred normalize False
2
```

In the multilabel case with binary label indicators

```
import numpy as np
accuracyscorenparray0 1 1 1 np.ones2 2
05
```

Examples using sklearn.metrics.accuracyscore

- Plot classification probability
- Multiclass AdaBoosted Decision Trees
- Probabilistic predictions with Gaussian process classification GPC
- Demonstration of multimetric evaluation on crossvalscore and GridSearchCV
- Importance of Feature Scaling
- Classification of text documents using sparse features

```
sklearn.metrics auc
sklearn.metrics aucxyreorder'deprecated'
```

Compute Area Under the Curve AUC using the trapezoidal rule

This is a general function given points on a curve For computing the area under the ROC curve see rocaucscore For an alternative way to summarize a precisionrecall curve see averageprecisionscore

Parameters

xarray shape n x coordinates These must be either monotonic increasing or monotonic decreasing
yarray shape n y coordinates
reorder boolean optional default'deprecated' Whether to sort x before computing If False assume that x must be either monotonic increasing or monotonic decreasing If True y is used to break ties when sorting x Make sure that y has a monotonic relation to x when setting reorder to True

Deprecated since version 020 Parameter reorder has been deprecated in version 020 and will be removed in 022 It's introduced for rocaucscore not for general use and is 624sklearn.metrics Metrics 1987

scikitlearn user guide Release 0213

no longer used there What’s more the result from auc will be significantly influenced if x is sorted unexpectedly due to slight floating point error See issue 9786 Future and default behavior is equivalent to reorderFalse

Returns

auc float

See also

rocaucscore Compute the area under the ROC curve

averageprecisionscore Compute average precision from prediction scores

precisionrecallcurve Compute precisionrecall pairs for different probability thresholds

Examples

```
import numpy as np
from sklearn import metrics
y nparray1 1 2 2
pred nparray01 04 035 08
fpr tpr thresholds metricsroccurve pred poslabel2
metricsaucfpr tpr
075
```

Examples using sklearnmetricsauc

- Species distribution modeling
- Receiver Operating Characteristic ROC with cross validation
- Receiver Operating Characteristic ROC

sklearnmetrics averageprecisionscore

sklearnmetrics averageprecisionscore ytrue yscore average’macro’ poslabel1

sampleweightNone

Compute average precision AP from prediction scores

AP summarizes a precisionrecall curve as the weighted mean of precisions achieved at each threshold with the increase in recall from the previous threshold used as the weight

APΣ

precisionrecall−1

where precisionandrecallare the precision and recall at the nth threshold 1 This implementation is not interpolated and is different from computing the area under the precisionrecall curve with the trapezoidal rule which uses linear interpolation and can be too optimistic

Note this implementation is restricted to the binary classification task or multilabel classification task

Read more in the User Guide

Parameters

ytrue array shape nsamples or nsamples nclasses True binary labels or binary label indicators

1988 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

yscore array shape nsamples or nsamples nclasses Target scores can either be probability estimates of the positive class confidence values or nonthresholded measure of decisions as returned by “decisionfunction” on some classifiers
average string None ‘micro’ ‘macro’ default ‘samples’ ‘weighted’ If None the scores for each class are returned Otherwise this determines the type of averaging performed on the data

micro Calculate metrics globally by considering each element of the label indicator matrix as a label

macro Calculate metrics for each label and find their unweighted mean This does not take label imbalance into account

weighted Calculate metrics for each label and find their average weighted by support the number of true instances for each label

samples Calculate metrics for each instance and find their average

Will be ignored when ytrue is binary

poslabel int or str default1 The label of the positive class Only applied to binary

ytrue For multilabelindicator ytrue poslabel is fixed to 1

sampleweight arraylike of shape nsamples optional Sample weights

Returns

averageprecision float

See also

rocaucscore Compute the area under the ROC curve

precisionrecallcurve Compute precisionrecall pairs for different probability thresholds

Notes

Changed in version 019 Instead of linearly interpolating between operating points precisions are weighted by the change in recall since the last operating point

References

1

Examples

```
import numpy as np
from sklearnmetrics import averageprecisionscore
ytrue nparray0 0 1 1
yscores nparray01 04 035 08
averageprecisionscoreytrue yscores
083
624sklearnmetrics Metrics 1989
```

scikitlearn user guide Release 0213

Examples using sklearnmetricsaverageprecisionscore

•PrecisionRecall

sklearnmetrics balancedaccuracyscore

sklearnmetrics balancedaccuracyscore ytrue ypred sampleweightNone ad

justedFalse

Compute the balanced accuracy

The balanced accuracy in binary and multiclass classification problems to deal with imbalanced datasets It is defined as the average of recall obtained on each class

The best value is 1 and the worst value is 0 when adjustedFalse

Read more in the User Guide

Parameters

ytrue 1d arraylike Ground truth correct target values

ypred 1d arraylike Estimated targets as returned by a classifier

sampleweight arraylike of shape nsamples optional Sample weights

adjusted bool defaultFalse When true the result is adjusted for chance so that random performance would score 0 and perfect performance scores 1

Returns

balancedaccuracy float

See also

recallscore rocaucscore

Notes

Some literature promotes alternative definitions of balanced accuracy Our definition is equivalent to accuracyscore with classbalanced sample weights and shares desirable properties with the binary case

See the User Guide

References

12

Examples

from sklearnmetrics import balancedaccuracyscore

ytrue 0 1 0 0 1 0

ypred 0 1 0 0 0 1

balancedaccuracyscoreytrue ypred

0625

1990 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearnmetrics brierscoreloss

sklearnmetrics brierscoreloss ytrue yprob sampleweightNone poslabelNone

Compute the Brier score The smaller the Brier score the better hence the naming with “loss” Across all items in a set N predictions the Brier score measures the mean squared difference between 1 the predicted probability assigned to the possible outcomes for item i and 2 the actual outcome Therefore the lower the Brier score is for a set of predictions the better the predictions are calibrated Note that the Brier score always takes on a value between zero and one since this is the largest possible difference between a predicted probability which must be between zero and one and the actual outcome which can take on values of only 0 and 1 The Brier loss is composed of refinement loss and calibration loss The Brier score is appropriate for binary and categorical outcomes that can be structured as true or false but is inappropriate for ordinal variables which can take on three or more values this is because the Brier score assumes that all possible outcomes are equivalently “distant” from one another Which label is considered to be the positive label is controlled via the parameter poslabel which defaults to 1 Read more in the User Guide

Parameters

ytrue array shape nsamples True targets

yprob array shape nsamples Probabilities of the positive class

sampleweight arraylike of shape nsamples optional Sample weights

poslabel int or str defaultNone Label of the positive class Defaults to the greater label

unless ytrue is all 0 or all 1 in which case poslabel defaults to 1

Returns

score float Brier score

References

1

Examples

import numpy as np

from sklearnmetrics import brierscoreloss

ytrue nparray0 1 1 0

ytruecategorical nparrayspam ham ham spam

yprob nparray01 09 08 03

brierscorelossytrue yprob

0037

brierscorelossytrue 1yprob poslabel0

0037

brierscorelossytruecategorical yprob pos

↪labelham

0037

brierscorelossytrue nparrayyprob 05

00

Examples using sklearnmetricsbrierscoreloss

•Probability Calibration curves

624sklearnmetrics Metrics 1991

scikitlearn user guide Release 0213

- Probability calibration of classifiers

sklearnmetrics classificationreport
sklearnmetrics classificationreport ytrue ypred labelsNone targetnamesNone
sampleweightNone digits2 outputdictFalse
Build a text report showing the main classification metrics
Read more in the User Guide

Parameters

ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct target values

ypred 1d arraylike or label indicator array sparse matrix Estimated targets as returned by a classifier

labels array shape nlabels Optional list of label indices to include in the report

targetnames list of strings Optional display names matching the labels same order

sampleweight arraylike of shape nsamples optional Sample weights

digits int Number of digits for formatting output floating point values When outputdict isTrue this will be ignored and the returned values will not be rounded

outputdict bool default False If True return output as dict

Returns

report string dict Text summary of the precision recall F1 score for each class Dictionary returned if outputdict is True Dictionary has the following structure

label 1 precision05
recall10
f1score067
support1
label 2

The reported averages include macro average averaging the unweighted mean per label weighted average averaging the supportweighted mean per label sample average only for multilabel classification and micro average averaging the to tal true positives false negatives and false positives it is only shown for multi label or multiclass with a subset of classes because it is accuracy otherwise See alsofuncprecisionrecallfscoresupport for more details on averages

Note that in binary classification recall of the positive class is also known as “sensitivity” recall of the negative class is “specificity”

See also

precisionrecallfscoresupport confusionmatrix
multilabelconfusionmatrix

1992 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

```
from sklearn.metrics import classification_report
ytrue 0 1 2 2 2
ypred 0 0 2 2 1
targetnames class 0 class 1 class 2
print(classification_report(ytrue, ypred, targetnames=targetnames))
precision recall f1score support
class 0 0.50 1.00 0.67 1
class 1 0.00 0.00 0.00 1
class 2 1.00 0.67 0.80 3
accuracy 0.60 5
macro avg 0.50 0.56 0.49 5
weighted avg 0.70 0.60 0.61 5
ypred 1 1 0
ytrue 1 1 1
print(classification_report(ytrue, ypred, labels=1, 2, 3))
precision recall f1score support
1 1.00 0.67 0.80 3
2 0.00 0.00 0.00 0
3 0.00 0.00 0.00 0
micro avg 1.00 0.67 0.80 3
macro avg 0.33 0.22 0.27 3
weighted avg 1.00 0.67 0.80 3
Examples using sklearn.metrics.classification_report
•Faces recognition example using eigenfaces and SVMs
•Recognizing handwritten digits
•Column Transformer with Heterogeneous Data Sources
•Pipeline Anova SVM (Support Vector Machine)
•Parameter estimation using grid search with crossvalidation
•Restricted Boltzmann Machine features for digit classification
•Label Propagation digits Demonstrating performance
•Label Propagation digits active learning
•Classification of text documents using sparse features
sklearn.metrics.cohenkappascore
sklearn.metrics.cohenkappascore y1 y2 labels=None weights=None sample_weight=None
Cohen's kappa a statistic that measures interannotator agreement
624sklearn.metrics Metrics 1993
```

scikitlearn user guide Release 0213

This function computes Cohen’s kappa 1 a score that expresses the level of agreement between two annotators on a classification problem It is defined as

$$\kappa = \frac{p - p_e}{1 - p_e}$$

where p is the empirical probability of agreement on the label assigned to any sample the observed agreement ratio and p_e is the expected agreement when both annotators assign labels randomly p_e is estimated using a perannotator empirical prior over the class labels 2

Read more in the User Guide

Parameters

y1array shape nsamples Labels assigned by the first annotator
y2array shape nsamples Labels assigned by the second annotator The kappa statistic is symmetric so swapping y1andy2doesn’t change the value
labels array shape nclasses optional List of labels to index the matrix This may be used to select a subset of labels If None all labels that appear at least once in y1ory2are used
weights str optional List of weighting type to calculate the score None means no weighted “linear” means linear weighted “quadratic” means quadratic weighted
sampleweight arraylike of shape nsamples optional Sample weights

Returns

kappa float The kappa statistic which is a number between -1 and 1 The maximum value means complete agreement zero or lower means chance agreement

References

123

sklearnmetrics confusionmatrix

sklearnmetrics confusionmatrix ytrue ypred labelsNone sampleweightNone

Compute confusion matrix to evaluate the accuracy of a classification

By definition a confusion matrix C is such that C_{ij} is equal to the number of observations known to be in group j but predicted to be in group i

Thus in binary classification the count of true negatives is C_{00} false negatives is C_{10} true positives is C_{11} and false positives is C_{01}

Read more in the User Guide

Parameters

ytrue array shape nsamples Ground truth correct target values
ypred array shape nsamples Estimated targets as returned by a classifier
labels array shape nclasses optional List of labels to index the matrix This may be used to reorder or select a subset of labels If none is given those that appear at least once in ytrue orypred are used in sorted order
sampleweight arraylike of shape nsamples optional Sample weights

Returns

Carray shape nclasses nclasses Confusion matrix

1994 Chapter 6 API (Application Programming Interface) Reference

References

1

Examples

```
from sklearnmetrics import confusionmatrix
ytrue 2 0 2 2 0 1
ypred 0 0 2 2 0 2
confusionmatrixytrue ypred
array2 0 0
0 0 1
1 0 2
```

```
ytrue cat ant cat cat ant bird
ypred ant ant cat cat ant cat
confusionmatrixytrue ypred labelsant bird cat
array2 0 0
0 0 1
1 0 2
```

In the binary case we can extract true positives etc as follows

```
tn fp fn tp confusionmatrix0 1 0 1 1 1 1 0
tn fp fn tp
0 2 1 1
```

Examples using sklearnmetricsconfusionmatrix

- Faces recognition example using eigenfaces and SVMs
- Recognizing handwritten digits
- Confusion matrix
- Label Propagation digits Demonstrating performance
- Label Propagation digits active learning
- Classification of text documents using sparse features

```
sklearnmetrics f1score
sklearnmetrics f1score ytrue ypred labelsNone poslabel1 average'binary' sam
pleweightNone
```

Compute the F1 score also known as balanced Fscore or Fmeasure

The F1 score can be interpreted as a weighted average of the precision and recall where an F1 score reaches its best value at 1 and worst score at 0 The relative contribution of precision and recall to the F1 score are equal

The formula for the F1 score is

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

scikitlearn user guide Release 0213

In the multiclass and multilabel case this is the average of the F1 score of each class with weighting depending on the average parameter

Read more in the User Guide

Parameters

ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct target values

ypred 1d arraylike or label indicator array sparse matrix Estimated targets as returned by a classifier

labels list optional The set of labels to include when average binary and their order if average is None Labels present in the data can be excluded for example to calculate a multiclass average ignoring a majority negative class while labels not present in the data will result in 0 components in a macro average For multilabel targets labels are column indices By default all labels in ytrue and ypred are used in sorted order Changed in version 0.17 parameter labels improved for multiclass problem poslabel str or int 1 by default The class to report if average binary and the data is binary If the data are multiclass or multilabel this will be ignored setting labels_poslabel and average binary will report scores for that label only

average string None 'binary' default 'micro' 'macro' 'samples' 'weighted' This parameter is required for multiclassmultilabel targets If None the scores for each class are returned Otherwise this determines the type of averaging performed on the data binary Only report results for the class specified by poslabel This is applicable only if targets ytrue ypred are binary

micro Calculate metrics globally by counting the total true positives false negatives and false positives

macro Calculate metrics for each label and find their unweighted mean This does not take label imbalance into account

weighted Calculate metrics for each label and find their average weighted by support the number of true instances for each label This alters 'macro' to account for label imbalance it can result in an Fscore that is not between precision and recall

samples Calculate metrics for each instance and find their average only meaningful for multilabel classification where this differs from accuracy score

sampleweight arraylike of shape nsamples optional Sample weights

Returns

f1score float or array of float shape nunique labels F1 score of the positive class in binary classification or weighted average of the F1 scores of each class for the multiclass task

See also

fbeta score precision recall f scores support jaccard score

multilabel confusion matrix

1996 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Notes

When true positive false positive 0 or true positive false negative 0 fscore returns 0 and raises UndefinedMetricWarning

References

1

Examples

from sklearn.metrics import f1score

ytrue 0 1 2 0 1 2

ypred 0 2 1 0 0 1

f1score(ytrue, ypred, average='macro')

0.26

f1score(ytrue, ypred, average='micro')

0.33

f1score(ytrue, ypred, average='weighted')

0.26

f1score(ytrue, ypred, average=None)

array([0.8, 0.0])

Examples using sklearn.metrics.f1score

•Probability Calibration curves

sklearn.metrics.fbeta_score

sklearn.metrics.fbeta_score(ytrue, ypred, beta=1, labels=None, pos_label=1, average='binary')

sample_weight=None

Compute the Fbeta score

The Fbeta score is the weighted harmonic mean of precision and recall reaching its optimal value at 1 and its worst value at 0

The beta parameter determines the weight of recall in the combined score: beta = 1 lends more weight to precision while beta = 1 favors recall. beta = 0 considers only precision, beta = inf only recall.

Read more in the User Guide

Parameters

ytrue 1d array-like or label indicator array, sparse matrix Ground truth (correct) target values

ypred 1d array-like or label indicator array, sparse matrix Estimated targets as returned by a classifier

beta float Weight of precision in harmonic mean

labels list optional The set of labels to include when average = 'binary' and their

order if average is None. Labels present in the data can be excluded, for example to

calculate a multiclass average ignoring a majority negative class while labels not present in

624 sklearn.metrics Metrics 1997

scikitlearn user guide Release 0213

the data will result in 0 components in a macro average For multilabel targets labels are column indices By default all labels in ytrue andypred are used in sorted order

Changed in version 017 parameter labels improved for multiclass problem

poslabel str or int 1 by default The class to report if averagebinary and the data is binary If the data are multiclass or multilabel this will be ignored setting labelsposlabel andaverage binary will report scores for that label only

average string None 'binary' default 'micro' 'macro' 'samples' 'weighted' This parameter is required for multiclassmultilabel targets If None the scores for each class are returned Otherwise this determines the type of averaging performed on the data binary Only report results for the class specified by poslabel This is applicable only if targets ytrueypred are binary

micro Calculate metrics globally by counting the total true positives false negatives and false positives

macro Calculate metrics for each label and find their unweighted mean This does not take label imbalance into account

weighted Calculate metrics for each label and find their average weighted by support the number of true instances for each label This alters 'macro' to account for label imbalance it can result in an Fscore that is not between precision and recall

samples Calculate metrics for each instance and find their average only meaningful for multilabel classification where this differs from accuracyscore

sampleweight arraylike of shape nsamples optional Sample weights

Returns
fbetascore float if average is not None or array of float shape nuniquelabels Fbeta score of the positive class in binary classification or weighted average of the Fbeta score of each class for the multiclass task

See also
precisionrecallfscoresupport multilabelconfusionmatrix

Notes
When true positive false positive 0 or true positive false negative 0 fscore returns 0 and raises UndefinedMetricWarning

References

12
Examples
from sklearnmetrics import fbeta
ytrue 0 1 2 0 1 2

ypred 0 2 1 0 0 1
fbetascoreytrue ypred averagemacro beta05

1998 Chapter 6 API (Application Programming Interface) Reference

023
fbetascoreytrue ypred averagemicro beta05

033
fbetascoreytrue ypred averageweighted beta05

023
fbetascoreytrue ypred average None beta05

array071 0 0
sklearnmetrics hammingloss
sklearnmetrics hammingloss ytrue ypred labelsNone sampleweightNone
Compute the average Hamming loss
The Hamming loss is the fraction of labels that are incorrectly predicted
Read more in the User Guide
Parameters
ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct labels
ypred 1d arraylike or label indicator array sparse matrix Predicted labels as returned by
a classifier
labels array shape nlabels optional default'deprecated' Integer array of labels If not
provided labels will be inferred from ytrue and ypred
New in version 018
Deprecated since version 021 This parameter labels is deprecated in version 021 and
will be removed in version 023 Hamming loss uses ytrueshape1 for the number
of labels when ytrue is binary label indicators so it is unnecessary for the user to specify
sampleweight arraylike of shape nsamples optional Sample weights
New in version 018
Returns
loss float or int Return the average Hamming loss between element of ytrue andypred
See also
accuracy score jaccardscore zeroone loss
Notes
In multiclass classification the Hamming loss corresponds to the Hamming distance between ytrue and
ypred which is equivalent to the subset zeroone loss function when normalize parameter is set to
True
In multilabel classification the Hamming loss is different from the subset zeroone loss The zeroone loss
considers the entire set of labels for a given sample incorrect if it does not entirely match the true set of labels
Hamming loss is more forgiving in that it penalizes only the individual labels
624sklearnmetrics Metrics 1999

scikitlearn user guide Release 0.21.3

The Hamming loss is upperbounded by the subset zero-one loss when normalize parameter is set to True. It is always between 0 and 1, lower being better.

References

12

Examples

```
from sklearn.metrics import hamming_loss
```

```
y_pred = [1, 2, 3, 4]
```

```
y_true = [2, 2, 3, 4]
```

```
hamming_loss(y_true, y_pred)
```

0.25

In the multilabel case with binary label indicators

```
import numpy as np
```

```
hamming_loss(np.array([0, 1, 1, 1]), np.zeros(2, 2))
```

0.75

Examples using sklearn.metrics.hamming_loss

•Model Complexity Influence

```
sklearn.metrics.hinge_loss
```

```
sklearn.metrics.hinge_loss(y_true, pred_decision, labels=None, sample_weight=None)
```

Average hinge loss, nonregularized

In binary class case assuming labels in y_true are encoded with 1 and -1 when a prediction mistake is

made, margin = y_true * pred_decision is always negative since the signs disagree, implying

margin is always greater than -1. The cumulated hinge loss is therefore an upper bound of the number of mistakes made by the classifier.

In multiclass case the function expects that either all the labels are included in y_true or an optional labels

argument is provided which contains all the labels. The multilabel margin is calculated according to Crammer

Singer's method. As in the binary case the cumulated hinge loss is an upper bound of the number of mistakes

made by the classifier.

Read more in the User Guide

Parameters

y_true: array, shape (n_samples,) True target consisting of integers of two values. The

positive label must be greater than the negative label.

pred_decision: array, shape (n_samples,) or (n_samples, n_classes). Predicted decisions as

output by decision function. floats.

labels: array, optional, default None. Contains all the labels for the problem. Used in multiclass

hinge loss.

sample_weight: array-like of shape (n_samples,) optional. Sample weights.

2000 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

loss float

References

123

Examples

```
from sklearn import svm
from sklearnmetrics import hingeloss
```

X 0 1

y 1 1

```
est svmLinearSVCrandomstate0
```

```
estfitX y
```

```
LinearSVCC10 classweightNone dualTrue fitinterceptTrue
```

```
interceptscaling1 losssquaredhinge maxiter1000
```

```
multiclassovr penaltyl2 randomstate0 tol00001
```

```
verbose0
```

```
preddecision estdecisionfunction2 3 05
```

```
preddecision
```

```
array218 236 009
```

```
hingeloss1 1 1 preddecision
```

030

In the multiclass case

```
import numpy as np
```

```
X nparray0 1 2 3
```

```
Y nparray0 1 2 3
```

```
labels nparray0 1 2 3
```

```
est svmLinearSVC
```

```
estfitX Y
```

```
LinearSVCC10 classweightNone dualTrue fitinterceptTrue
```

```
interceptscaling1 losssquaredhinge maxiter1000
```

```
multiclassovr penaltyl2 randomstateNone tol00001
```

```
verbose0
```

```
preddecision estdecisionfunction1 2 3
```

```
ytrue 0 2 3
```

```
hingelossytrue preddecision labels
```

056

```
sklearnmetrics jaccardscore
```

```
sklearnmetrics jaccardscore ytrue ypred labelsNone poslabel1 average'binary'
```

```
sampleweightNone
```

Jaccard similarity coefficient score

The Jaccard index 1 or Jaccard similarity coefficient defined as the size of the intersection divided by the size of the union of two label sets is used to compare set of predicted labels for a sample to the corresponding set of labels in ytrue

Read more in the User Guide

624sklearnmetrics Metrics 2001

scikitlearn user guide Release 0213

Parameters

ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct labels
ypred 1d arraylike or label indicator array sparse matrix Predicted labels as returned by
a classifier

labels list optional The set of labels to include when average binary and their
order ifaverage is None Labels present in the data can be excluded for example to
calculate a multiclass average ignoring a majority negative class while labels not present in
the data will result in 0 components in a macro average For multilabel targets labels are
column indices By default all labels in ytrue andypred are used in sorted order
poslabel str or int 1 by default The class to report if averagebinary and the
data is binary If the data are multiclass or multilabel this will be ignored setting
labelsposlabel andaverage binary will report scores for that la
bel only

average string None 'binary' default 'micro' 'macro' 'samples' 'weighted' If None
the scores for each class are returned Otherwise this determines the type of averaging
performed on the data

binary Only report results for the class specified by poslabel This is applicable
only if targets ytruepred are binary

micro Calculate metrics globally by counting the total true positives false negatives
and false positives

macro Calculate metrics for each label and find their unweighted mean This does not
take label imbalance into account

weighted Calculate metrics for each label and find their average weighted by sup
port the number of true instances for each label This alters 'macro' to account for label
imbalance

samples Calculate metrics for each instance and find their average only meaningful
for multilabel classification

sampleweight arraylike of shape nsamples optional Sample weights

Returns

score float if average is not None or array of floats shape nuniquelabels

See also

accuracy score fscore multilabelconfusionmatrix

Notes

jaccardscore may be a poor metric if there are no positives for some samples or classes Jaccard is
undefined if there are no true or predicted labels and our implementation will return a score of 0 with a warning

References

1

2002 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

```
import numpy as np
from sklearnmetrics import jaccardscore
ytrue nparray0 1 1
1 1 0
ypred nparray1 1 1
1 0 0
```

In the binary case

```
jaccardscoreytrue0 ypred0
06666
```

In the multilabel case

```
jaccardscoreytrue ypred averagesamples
05833
```

```
jaccardscoreytrue ypred averagemacro
06666
```

```
jaccardscoreytrue ypred average None
array05 05 1
```

In the multiclass case

```
ypred 0 2 1 2
ytrue 0 1 2 2
```

```
jaccardscoreytrue ypred average None
```

```
array1 0 033
```

Examples using sklearnmetricsjaccardscore

•Classifier Chain

sklearnmetrics logloss

```
sklearnmetrics logloss ytrue ypred eps1e15 normalizeTrue sampleweightNone la
belsNone
```

Log loss aka logistic loss or crossentropy loss

This is the loss function used in multinomial logistic regression and extensions of it such as neural networks defined as the negative loglikelihood of the true labels given a probabilistic classifier’s predictions The log loss is only defined for two or more labels For a single sample with true label yt in 01 and estimated probability

yp that yt 1 the log loss is

```
log Pytyp yt logyp 1 yt log1 yp
```

Read more in the User Guide

Parameters

ytrue arraylike or label indicator matrix Ground truth correct labels for nsamples sam
ples

scikitlearn user guide Release 0213

ypred arraylike of float shape nsamples nclasses or nsamples Predicted probabilities as returned by a classifier's predict_proba method If ypredshape nsamples the probabilities provided are assumed to be that of the positive class The labels in ypred are assumed to be ordered alphabetically as done by preprocessing LabelBinarizer

eps float Log loss is undefined for p0 or p1 so probabilities are clipped to maxeps min1 eps p

normalize bool optional default True If true return the mean loss per sample Otherwise return the sum of the per sample losses

sampleweight arraylike of shape nsamples optional Sample weights

labels arraylike optional default None If not provided labels will be inferred from ytrue If labels is None and ypred has shape nsamples the labels are assumed to be binary and are inferred from ytrue version added 018

Returns

loss float

Notes

The logarithm used is the natural logarithm base e

References

CM Bishop 2006 Pattern Recognition and Machine Learning Springer p 209

Examples

```
from sklearn.metrics import log_loss
log_loss(spam, ham, ham, spam)
1.991823565021616
```

Examples using sklearn.metrics.log_loss

- Probability Calibration for 3class classification
- Probabilistic predictions with Gaussian process classification GPC

```
sklearn.metrics.matthews_corrcoef
sklearn.metrics.matthews_corrcoef(ytrue, ypred, sample_weight=None)
```

Compute the Matthews correlation coefficient MCC

The Matthews correlation coefficient is used in machine learning as a measure of the quality of binary and multiclass classifications It takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes The MCC is in essence a correlation coefficient value between -1 and 1 A coefficient of 1 represents a perfect prediction 0 an average

2004 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

random prediction and 1 an inverse prediction The statistic is also known as the phi coefficient source
Wikipedia

Binary and multiclass labels are supported Only in the binary case does this relate to information about true
and false positives and negatives See references below

Read more in the User Guide

Parameters

ytrue array shape nsamples Ground truth correct target values

ypred array shape nsamples Estimated targets as returned by a classifier

sampleweight arraylike of shape nsamples default None Sample weights

Returns

mcc float The Matthews correlation coefficient 1 represents a perfect prediction 0 an aver

age random prediction and 1 and inverse prediction

References

1234

Examples

```
from sklearnmetrics import matthewscorrcoef
```

```
ytrue 1 1 1 1
```

```
ypred 1 1 1 1
```

```
matthewscorrcoefytrue ypred
```

```
033
```

```
sklearnmetrics multilabelconfusionmatrix
```

```
sklearnmetrics multilabelconfusionmatrix ytrue ypred sampleweightNone la
```

```
belsNone samplewiseFalse
```

Compute a confusion matrix for each class or sample

New in version 021

Compute classwise default or samplewise samplewiseTrue multilabel confusion matrix to evaluate the

accuracy of a classification and output confusion matrices for each class or sample

In multilabel confusion matrix `[[[0]]]` the count of true negatives is `[[[0]]]` 00 false negatives is `[[[0]]]` 10

true positives is `[[[0]]]` 11and false positives is `[[[0]]]` 01

Multiclass data will be treated as if binarized under a onevsrest transformation Returned confusion matrices

will be in the order of sorted unique labels in the union of ytrue ypred

Read more in the User Guide

Parameters

ytrue 1d arraylike or label indicator array sparse matrix of shape nsamples noutputs

or nsamples Ground truth correct target values

ypred 1d arraylike or label indicator array sparse matrix of shape nsamples noutputs

or nsamples Estimated targets as returned by a classifier

624sklearnmetrics Metrics 2005

scikitlearn user guide Release 0213

sampleweight arraylike of shape nsamples optional Sample weights
labels arraylike A list of classes or column indices to select some or to force inclusion of
classes absent from the data
samplewise bool defaultFalse In the multilabel case this calculates a confusion matrix per
sample

Returns
multiconfusion array shape noutputs 2 2 A 2x2 confusion matrix corresponding to each
output in the input When calculating classwise multiconfusion default then noutputs
nlabels when calculating samplewise multiconfusion samplewiseTrue noutputs
nsamples If labels is defined the results will be returned in the order specified in
labels otherwise the results will be returned in sorted order by default

See also
confusionmatrix
Notes

The multilabelconfusionmatrix calculates classwise or samplewise multilabel confusion matrices and in
multiclass tasks labels are binarized under a onevsrest way while confusionmatrix calculates one confusion
matrix for confusion between every two classes

Examples
Multilabelindicator case
import numpy as np
from sklearnmetrics import multilabelconfusionmatrix
ytrue nparray1 0 1
0 1 0
ypred nparray1 0 0
0 1 1
multilabelconfusionmatrixytrue ypred
array1 0
0 1
1 0
0 1
0 1
1 0

Multiclass case
ytrue cat ant cat cat ant bird
ypred ant ant cat cat ant cat
multilabelconfusionmatrixytrue ypred
labelsant bird cat
array3 1
0 2
5 0
1 0

scikitlearn user guide Release 0213

2 1
1 2

sklearnmetrics precisionrecallcurve

sklearnmetrics precisionrecallcurve ytrue probaspred poslabelNone sam
pleweightNone

Compute precisionrecall pairs for different probability thresholds

Note this implementation is restricted to the binary classification task

The precision is the ratio $\frac{tp}{tp + fp}$ where tp is the number of true positives and fp the number of false positives The precision is intuitively the ability of the classifier not to label as positive a sample that is negative

The recall is the ratio $\frac{tp}{tp + fn}$ where tp is the number of true positives and fn the number of false negatives The recall is intuitively the ability of the classifier to find all the positive samples

The last precision and recall values are 1 and 0 respectively and do not have a corresponding threshold This ensures that the graph starts on the y axis

Read more in the User Guide

Parameters

ytrue array shape nsamples True binary labels If labels are not either 1 1 or 0 1 then poslabel should be explicitly given

probaspred array shape nsamples Estimated probabilities or decision function
poslabel int or str defaultNone The label of the positive class When poslabelNone

if ytrue is in 1 1 or 0 1 poslabel is set to 1 otherwise an error will be raised
sampleweight arraylike of shape nsamples optional Sample weights

Returns

precision array shape nthresholds 1 Precision values such that element i is the precision of predictions with score thresholds[i] and the last element is 1

recall array shape nthresholds 1 Decreasing recall values such that element i is the recall of predictions with score thresholds[i] and the last element is 0

thresholds array shape nthresholds lennpuniqueprobaspred Increasing thresholds on the decision function used to compute precision and recall

See also

averageprecisionscore Compute average precision from prediction scores

roccurve Compute Receiver operating characteristic ROC curve

Examples

```
import numpy as np
from sklearnmetrics import precisionrecallcurve
ytrue nparray0 0 1 1
yscores nparray01 04 035 08
624sklearnmetrics Metrics 2007
```

scikitlearn user guide Release 0213

precision recall thresholds precisionrecallcurve

ytrue yscores

precision

array066666667 05 1 1

recall

array1 05 05 0

thresholds

array035 04 08

Examples using sklearnmetricsprecisionrecallcurve

- PrecisionRecall

sklearnmetrics precisionrecallfscoresupport

sklearnmetrics precisionrecallfscoresupport ytrue ypred beta10 la

belsNone poslabel1 aver

ageNone warnfor'precision'

'recall' 'fscore' sam

pleweightNone

Compute precision recall Fmeasure and support for each class

The precision is the ratio $tp / (tp + fp)$ wheretpis the number of true positives and fpthe number of false positives The precision is intuitively the ability of the classifier not to label as positive a sample that is negative

The recall is the ratio $tp / (tp + fn)$ wheretpis the number of true positives and fnthe number of false negatives The recall is intuitively the ability of the classifier to find all the positive samples

The Fbeta score can be interpreted as a weighted harmonic mean of the precision and recall where an Fbeta score reaches its best value at 1 and worst score at 0

The Fbeta score weights recall more than precision by a factor of beta beta 10 means recall and precision are equally important

The support is the number of occurrences of each class in ytrue

Ifposlabel is None and in binary classification this function returns the average precision recall and Fmeasure if average is one ofmicro macro weighted orsamples

Read more in the User Guide

Parameters

ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct target values

ypred 1d arraylike or label indicator array sparse matrix Estimated targets as returned by a classifier

beta float 10 by default The strength of recall versus precision in the Fscore

labels list optional The set of labels to include when average binary and their order ifaverage is None Labels present in the data can be excluded for example to calculate a multiclass average ignoring a majority negative class while labels not present in the data will result in 0 components in a macro average For multilabel targets labels are column indices By default all labels in ytrue andypred are used in sorted order

2008 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

poslabel str or int 1 by default The class to report if averagebinary and the data is binary If the data are multiclass or multilabel this will be ignored setting labelsposlabel andaverage binary will report scores for that la

bel only

average string None default 'binary' 'micro' 'macro' 'samples' 'weighted' If None the scores for each class are returned Otherwise this determines the type of averaging performed on the data

binary Only report results for the class specified by poslabel This is applicable

only if targets ytruepred are binary

micro Calculate metrics globally by counting the total true positives false negatives and false positives

macro Calculate metrics for each label and find their unweighted mean This does not take label imbalance into account

weighted Calculate metrics for each label and find their average weighted by support the number of true instances for each label This alters 'macro' to account for label imbalance it can result in an Fscore that is not between precision and recall

samples Calculate metrics for each instance and find their average only meaningful for multilabel classification where this differs from accuracyscore

warnfor tuple or set for internal use This determines which warnings will be made in the case that this function is being used to return only one of its metrics

sampleweight arraylike of shape nsamples optional Sample weights

Returns

precision float if average is not None or array of float shape nuniquelabels

recall float if average is not None or array of float shape nuniquelabels

fbetascore float if average is not None or array of float shape nuniquelabels

support int if average is not None or array of int shape nuniquelabels The number of occurrences of each label in ytrue

Notes

Whenttrue positive false positive 0 precision is undefined When true positive

false negative 0 recall is undefined In such cases the metric will be set to 0 as will fscore

andUndefinedMetricWarning will be raised

References

123

Examples

```
import numpy as np
from sklearnmetrics import precisionrecallfscoresupport
ytrue nparraycat dog pig cat dog pig
ypred nparraycat pig dog cat cat dog
624sklearnmetrics Metrics 2009
```

scikitlearn user guide Release 0213
precisionrecallfscoresupportytrue ypred averagemacro

022 033 026 None
precisionrecallfscoresupportytrue ypred averagemicro

033 033 033 None
precisionrecallfscoresupportytrue ypred averageweighted

022 033 026 None
It is possible to compute perlabel precisions recalls F1scores and supports instead of averaging
precisionrecallfscoresupportytrue ypred average None
labelspig dog cat

array0 0 066
array0 0 1 array0 0 08
array2 2 2
sklearnmetrics precisionscore
sklearnmetrics precisionscore ytrue ypred labelsNone poslabel1 average'binary'
sampleweightNone
Compute the precision

The precision is the ratio $\frac{tp}{tp + fp}$ wheretpis the number of true positives and fpthe number of false positives The precision is intuitively the ability of the classifier not to label as positive a sample that is negative
The best value is 1 and the worst value is 0
Read more in the User Guide

Parameters
ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct target values
ypred 1d arraylike or label indicator array sparse matrix Estimated targets as returned by a classifier
labels list optional The set of labels to include when average binary and their order ifaverage is None Labels present in the data can be excluded for example to calculate a multiclass average ignoring a majority negative class while labels not present in the data will result in 0 components in a macro average For multilabel targets labels are column indices By default all labels in ytrue andypred are used in sorted order
Changed in version 017 parameter labels improved for multiclass problem
poslabel str or int 1 by default The class to report if averagebinary and the data is binary If the data are multiclass or multilabel this will be ignored setting labelsposlabel andaverage binary will report scores for that label only
average string None 'binary' default 'micro' 'macro' 'samples' 'weighted' This parameter is required for multiclassmultilabel targets If None the scores for each class are returned Otherwise this determines the type of averaging performed on the data
2010 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0.21.3

binary Only report results for the class specified by poslabel. This is applicable only if targets ytrue ypred are binary

micro Calculate metrics globally by counting the total true positives false negatives and false positives

macro Calculate metrics for each label and find their unweighted mean. This does not take label imbalance into account

weighted Calculate metrics for each label and find their average weighted by support the number of true instances for each label. This alters 'macro' to account for label imbalance it can result in an Fscore that is not between precision and recall

samples Calculate metrics for each instance and find their average only meaningful for multilabel classification where this differs from accuracyscore

sampleweight arraylike of shape nsamples optional Sample weights

Returns

precision float if average is not None or array of float shape nunique labels Precision of the positive class in binary classification or weighted average of the precision of each class for the multiclass task

See also

precisionrecallfscoresupport multilabelconfusionmatrix

Notes

When true positive false positive = 0 precision returns 0 and raises UndefinedMetricWarning

Examples

```
from sklearn.metrics import precision_score
ytrue = [0, 1, 2, 0, 1, 2]
ypred = [0, 2, 1, 0, 0, 1]
precision_score(ytrue, ypred, average='macro')
0.22
precision_score(ytrue, ypred, average='micro')
0.33
precision_score(ytrue, ypred, average='weighted')
```

0.22

```
precision_score(ytrue, ypred, average=None)
array([0.66, 0. , 0. ])
```

Examples using sklearn.metrics.precision_score

- Probability Calibration curves

624sklearn.metrics Metrics 2011

scikitlearn user guide Release 0213

sklearnmetrics recallscore

sklearnmetrics recallscore ytrue ypred labelsNone poslabel1 average'binary' sampleweightNone

Compute the recall

The recall is the ratio $\frac{tp}{tp + fn}$ where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples. The best value is 1 and the worst value is 0.

Read more in the User Guide

Parameters

ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct target values

ypred 1d arraylike or label indicator array sparse matrix Estimated targets as returned by a classifier

labels list optional The set of labels to include when average 'binary' and their order if average is None. Labels present in the data can be excluded for example to calculate a multiclass average ignoring a majority negative class while labels not present in the data will result in 0 components in a macro average. For multilabel targets labels are column indices. By default all labels in ytrue and ypred are used in sorted order.

Changed in version 0.17 parameter labels improved for multiclass problem.
poslabel str or int 1 by default The class to report if average 'binary' and the data is binary. If the data are multiclass or multilabel this will be ignored setting labelsposlabel and average 'binary' will report scores for that label only.

average string None 'binary' default 'micro' 'macro' 'samples' 'weighted' This parameter is required for multiclass/multilabel targets. If None the scores for each class are returned. Otherwise this determines the type of averaging performed on the data: 'binary' Only report results for the class specified by poslabel. This is applicable only if targets ytrue/ypred are binary.

'micro' Calculate metrics globally by counting the total true positives, false negatives and false positives.

'macro' Calculate metrics for each label and find their unweighted mean. This does not take label imbalance into account.

'weighted' Calculate metrics for each label and find their average weighted by support the number of true instances for each label. This alters 'macro' to account for label imbalance. It can result in an F-score that is not between precision and recall.

'samples' Calculate metrics for each instance and find their average (only meaningful for multilabel classification where this differs from accuracy score).

sampleweight arraylike of shape (n_samples,) optional Sample weights

Returns

recall float if average is not None or array of float shape (n_unique_labels,) Recall of the positive class in binary classification or weighted average of the recall of each class for the multiclass task.

See also

2012 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

precisionrecallfscoresupport balancedaccuracy
score
multilabelconfusionmatrix

Notes

When true positive false negative 0 recall returns 0 and raises

UndefinedMetricWarning

Examples

```
from sklearn.metrics import recall_score
```

```
y_true = [0, 1, 2, 0, 1, 2]
```

```
y_pred = [0, 2, 1, 0, 0, 1]
```

```
recall_score(y_true, y_pred, average='macro')
```

```
0.33
```

```
recall_score(y_true, y_pred, average='micro')
```

```
0.33
```

```
recall_score(y_true, y_pred, average='weighted')
```

```
0.33
```

```
recall_score(y_true, y_pred, average=None)
```

```
array([1, 0, 0])
```

Examples using sklearn.metrics.recall_score

- Probability Calibration curves

```
sklearn.metrics.roc_auc_score
```

```
sklearn.metrics.roc_auc_score(y_true, y_score, average='macro', sample_weight=None)
```

```
max_fpr=None
```

Compute Area Under the Receiver Operating Characteristic Curve ROC AUC from prediction scores

Note this implementation is restricted to the binary classification task or multilabel classification task in label

indicator format

Read more in the User Guide

Parameters

y_true array shape (n_samples) or (n_samples, n_classes) True binary labels or binary label

indicators

y_score array shape (n_samples) or (n_samples, n_classes) Target scores can either be probability estimates of the positive class confidence values or nonthresholded measure of decisions as returned by “decision_function” on some classifiers For binary y_true

y_score is supposed to be the score of the class with greater label

average string None ‘micro’ ‘macro’ default ‘samples’ ‘weighted’ If None the scores

for each class are returned Otherwise this determines the type of averaging performed on the data

micro Calculate metrics globally by considering each element of the label indicator matrix as a label

624sklearn.metrics Metrics 2013

scikitlearn user guide Release 0.21.3

macro Calculate metrics for each label and find their unweighted mean. This does not take label imbalance into account.

weighted Calculate metrics for each label and find their average weighted by support the number of true instances for each label.

samples Calculate metrics for each instance and find their average.

Will be ignored when ytrue is binary.

sampleweight arraylike of shape (n_samples,) optional Sample weights

maxfpr float, 0 and 1 optional If not None, the standardized partial AUC over the range [0, maxfpr] is returned.

Returns

auc float

See also

average_precision_score Area under the precision-recall curve.

roc_curve Compute Receiver operating characteristic (ROC) curve.

References

123

Examples

```
import numpy as np
```

```
from sklearn.metrics import roc_auc_score
```

```
ytrue = np.array([0, 0, 1, 1])
```

```
yscores = np.array([0.1, 0.4, 0.35, 0.8])
```

```
roc_auc_score(ytrue, yscores)
```

```
0.75
```

```
sklearn.metrics.roc_curve
```

```
sklearn.metrics.roc_curve(ytrue, yscore, poslabel=None, sampleweight=None,
```

```
dropintermediate=True)
```

Compute Receiver operating characteristic (ROC)

Note this implementation is restricted to the binary classification task.

Read more in the User Guide.

Parameters

ytrue array, shape (n_samples,) True binary labels. If labels are not either 1 or 0, 1

then poslabel should be explicitly given.

yscore array, shape (n_samples,) Target scores can either be probability estimates of the positive class confidence values or nonthresholded measure of decisions as returned by "decisionfunction" on some classifiers.

poslabel int or str, default=None The label of the positive class. When poslabel=None

if ytrue is in [1, 1] or [0, 1] poslabel is set to 1, otherwise an error will be raised.

2014 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sampleweight arraylike of shape nsamples optional Sample weights
dropintermediate boolean optional defaultTrue Whether to drop some suboptimal
thresholds which would not appear on a plotted ROC curve This is useful in order to
create lighter ROC curves

New in version 017 parameter dropintermediate

Returns

fprarray shape 2 Increasing false positive rates such that element i is the false positive
rate of predictions with score thresholdsi

tprarray shape 2 Increasing true positive rates such that element i is the true positive
rate of predictions with score thresholdsi

thresholds array shape nthresholds Decreasing thresholds on the decision function used
to compute fpr and tpr thresholds0 represents no instances being predicted and is
arbitrarily set to maxyscore 1

See also

rocaucscore Compute the area under the ROC curve

Notes

Since the thresholds are sorted from low to high values they are reversed upon returning them to ensure they
correspond to both fpr andtpr which are sorted in reversed order during their calculation

References

12

Examples

```
import numpy as np
from sklearn import metrics
y nparray1 1 2 2
scores nparray01 04 035 08
fpr tpr thresholds metricsroccurve scores poslabel2
```

```
fpr
array0 0 05 05 1
```

```
tpr
array0 05 05 1 1
```

```
thresholds
array18 08 04 035 01
```

Examples using sklearnmetricsroccurve

- Species distribution modeling
 - Feature transformations with ensembles of trees
 - Receiver Operating Characteristic ROC with cross validation
- 624sklearnmetrics Metrics 2015

scikitlearn user guide Release 0213

•Receiver Operating Characteristic ROC

sklearnmetrics zerooneloss

sklearnmetrics zerooneloss ytrue ypred normalizeTrue sampleweightNone

Zeroone classification loss

If normalize is True return the fraction of misclassifications float else it returns the number of misclassifica

tions int The best performance is 0

Read more in the User Guide

Parameters

ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct labels

ypred 1d arraylike or label indicator array sparse matrix Predicted labels as returned by

a classifier

normalize bool optional defaultTrue If False return the number of misclassifications

Otherwise return the fraction of misclassifications

sampleweight arraylike of shape nsamples optional Sample weights

Returns

loss float or int If normalize True return the fraction of misclassifications float

else it returns the number of misclassifications int

See also

accuracy score hammingloss jaccardscore

Notes

In multilabel classification the zerooneloss function corresponds to the subset zeroone loss for each sample

the entire set of labels must be correctly predicted otherwise the loss for that sample is equal to one

Examples

```
from sklearnmetrics import zerooneloss
```

```
ytrue 1 2 3 4
```

```
ytrue 2 2 3 4
```

```
zeroonelossytrue ypred
```

```
025
```

```
zeroonelossytrue ypred normalize False
```

```
1
```

In the multilabel case with binary label indicators

```
import numpy as np
```

```
zeroonelossnparray0 1 1 1 np.ones2 2
```

```
05
```

2016 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples using sklearnmetricszerooneloss

- Discrete versus Real AdaBoost

6243 Regression metrics

See the Regression metrics section of the user guide for further details

metricsexplainedvariancescore ytrue

ypredExplained variance regression score function

metricsmaxerror ytrue ypred maxerror metric calculates the maximum residual error

metricsmeanabsoluteerror ytrue ypred Mean absolute error regression loss

metricsmmeansquarederror ytrue ypred

Mean squared error regression loss

metricsmmeansquaredlogerror ytrue

ypredMean squared logarithmic error regression loss

metricsmmedianabsoluteerror ytrue

ypredMedian absolute error regression loss

metricscr2score ytrue ypred R2 coefficient of determination regression score function

sklearnmetrics explainedvariancescore

sklearnmetrics explainedvariancescore ytrue ypred sampleweightNone multioutput

put'uniformaverage'

Explained variance regression score function

Best possible score is 10 lower values are worse

Read more in the User Guide

Parameters

ytrue arraylike of shape nsamples or nsamples noutputs Ground truth correct target values

ypred arraylike of shape nsamples or nsamples noutputs Estimated target values

sampleweight arraylike of shape nsamples optional Sample weights

multioutput string in 'rawvalues' 'uniformaverage' 'varianceweighted' or arraylike of shape noutputs Defines aggregating of multiple output scores Arraylike value defines weights used to average scores

'rawvalues' Returns a full set of scores in case of multioutput input

'uniformaverage' Scores of all outputs are averaged with uniform weight

'varianceweighted' Scores of all outputs are averaged weighted by the variances of each individual output

Returns

score float or ndarray of floats The explained variance or ndarray if 'multioutput' is 'rawvalues'

624sklearnmetrics Metrics 2017

scikitlearn user guide Release 0213

Notes

This is not a symmetric function

Examples

```
from sklearnmetrics import explainedvariancescore
ytrue 3 05 2 7
ypred 25 00 2 8
explainedvariancescoreytrue ypred
0957
ytrue 05 1 1 1 7 6
ypred 0 2 1 2 8 5
explainedvariancescoreytrue ypred multioutputuniformaverage
```

0983

sklearnmetrics maxerror

sklearnmetrics maxerror ytrue ypred

maxerror metric calculates the maximum residual error

Read more in the User Guide

Parameters

ytrue arraylike of shape nsamples Ground truth correct target values

ypred arraylike of shape nsamples Estimated target values

Returns

maxerror float A positive floating point value the best value is 00

Examples

```
from sklearnmetrics import maxerror
ytrue 3 2 7 1
ypred 4 2 7 1
maxerrorytrue ypred
1
sklearnmetrics meanabsoluteerror
sklearnmetrics meanabsoluteerror ytrue ypred sampleweightNone multiout
put'uniformaverage'
Mean absolute error regression loss
Read more in the User Guide
```

Parameters

ytrue arraylike of shape nsamples or nsamples noutputs Ground truth correct target values

2018 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

ypred arraylike of shape nsamples or nsamples noutputs Estimated target values

sampleweight arraylike of shape nsamples optional Sample weights

multioutput string in 'rawvalues' 'uniformaverage' or arraylike of shape noutputs

Defines aggregating of multiple output values Arraylike value defines weights used to average errors

'rawvalues' Returns a full set of errors in case of multioutput input

'uniformaverage' Errors of all outputs are averaged with uniform weight

Returns

loss float or ndarray of floats If multioutput is 'rawvalues' then mean absolute error is re

turned for each output separately If multioutput is 'uniformaverage' or an ndarray of

weights then the weighted average of all output errors is returned

MAE output is nonnegative floating point The best value is 00

Examples

from sklearnmetrics import meanabsoluteerror

ytrue 3 05 2 7

ypred 25 00 2 8

meanabsoluteerrorytrue ypred

05

ytrue 05 1 1 1 7 6

ypred 0 2 1 2 8 5

meanabsoluteerrorytrue ypred

075

meanabsoluteerrorytrue ypred multioutputrawvalues

array05 1

meanabsoluteerrorytrue ypred multioutput03 07

085

sklearnmetrics meansquarederror

sklearnmetrics meansquarederror ytrue ypred sampleweightNone multiout

put'uniformaverage'

Mean squared error regression loss

Read more in the User Guide

Parameters

ytrue arraylike of shape nsamples or nsamples noutputs Ground truth correct

target values

ypred arraylike of shape nsamples or nsamples noutputs Estimated target values

sampleweight arraylike of shape nsamples optional Sample weights

multioutput string in 'rawvalues' 'uniformaverage' or arraylike of shape noutputs

Defines aggregating of multiple output values Arraylike value defines weights used to average errors

'rawvalues' Returns a full set of errors in case of multioutput input

624sklearnmetrics Metrics 2019

scikitlearn user guide Release 0213

‘uniformaverage’ Errors of all outputs are averaged with uniform weight

Returns

loss float or ndarray of floats A nonnegative floating point value the best value is 00 or an array of floating point values one for each individual target

Examples

```
from sklearnmetrics import meansquarederror
ytrue 3 05 2 7
ypred 25 00 2 8
meansquarederrorytrue ypred
0375
ytrue 05 11 17 6
ypred 0 21 28 5
meansquarederrorytrue ypred
0708
meansquarederrorytrue ypred multioutputrawvalues
```

```
array041666667 1
meansquarederrorytrue ypred multioutput03 07
```

0825

Examples using sklearnmetricsmeansquarederror

- Model Complexity Influence
- Gradient Boosting regression
- Plot Ridge coefficients as a function of the L2 regularization
- Linear Regression Example
- Robust linear estimator fitting

```
sklearnmetrics meansquaredlogerror ytrue ypred sampleweightNone multiout
put‘uniformaverage’
```

Mean squared logarithmic error regression loss

Read more in the User Guide

Parameters

ytrue arraylike of shape nsamples or nsamples noutputs Ground truth correct target values
ypred arraylike of shape nsamples or nsamples noutputs Estimated target values
sampleweight arraylike of shape nsamples optional Sample weights
multioutput string in ‘rawvalues’ ‘uniformaverage’ or arraylike of shape noutputs
Defines aggregating of multiple output values Arraylike value defines weights used to average errors

2020 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

‘rawvalues’ Returns a full set of errors when the input is of multioutput format

‘uniformaverage’ Errors of all outputs are averaged with uniform weight

Returns

loss float or ndarray of floats A nonnegative floating point value the best value is 00 or an array of floating point values one for each individual target

Examples

```
from sklearn.metrics import meansquaredlogerror
ytrue 3 5 25 7
ypred 25 5 4 8
meansquaredlogerrorytrue ypred
0039
ytrue 05 1 1 2 7 6
ypred 05 2 1 25 8 8
meansquaredlogerrorytrue ypred
0044
meansquaredlogerrorytrue ypred multioutputrawvalues
```

array000462428 008377444
meansquaredlogerrorytrue ypred multioutput03 07

0060

sklearnmetrics medianabsoluteerror
sklearnmetrics medianabsoluteerror ytrue ypred
Median absolute error regression loss
Read more in the User Guide

Parameters

ytrue arraylike of shape nsamples Ground truth correct target values
ypred arraylike of shape nsamples Estimated target values

Returns

loss float A positive floating point value the best value is 00

Examples

```
from sklearn.metrics import medianabsoluteerror
ytrue 3 05 2 7
ypred 25 00 2 8
medianabsoluteerrorytrue ypred
05
```

Examples using sklearnmetricsmedianabsoluteerror
•Effect of transforming the targets in regression model
624sklearnmetrics Metrics 2021

scikitlearn user guide Release 0213

sklearnmetrics r2score

sklearnmetrics r2score ytrue ypred sampleweightNone multioutput'uniformaverage'

R2 coefficient of determination regression score function

Best possible score is 10 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 00

Read more in the User Guide

Parameters

ytrue arraylike of shape nsamples or nsamples noutputs Ground truth correct target values

ypred arraylike of shape nsamples or nsamples noutputs Estimated target values

sampleweight arraylike of shape nsamples optional Sample weights

multioutput string in 'rawvalues' 'uniformaverage' 'varianceweighted' or None or

arraylike of shape noutputs Defines aggregating of multiple output scores Arraylike

value defines weights used to average scores Default is "uniformaverage"

'rawvalues' Returns a full set of scores in case of multioutput input

'uniformaverage' Scores of all outputs are averaged with uniform weight

'varianceweighted' Scores of all outputs are averaged weighted by the variances of each individual output

Changed in version 019 Default value of multioutput is 'uniformaverage'

Returns

zfloat or ndarray of floats The R2 score or ndarray of scores if 'multioutput' is 'rawvalues'

Notes

This is not a symmetric function

Unlike most other scores R2 score may be negative it need not actually be the square of a quantity R

This metric is not welldefined for single samples and will return a NaN value if nsamples is less than two

References

1

Examples

from sklearnmetrics import r2score

ytrue 3 05 2 7

ypred 25 00 2 8

r2scoreytrue ypred

0948

ytrue 05 1 1 1 7 6

ypred 0 2 1 2 8 5

r2scoreytrue ypred

multioutputvarianceweighted

2022 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

0938

ytrue 1 2 3

ypred 1 2 3

r2scoreytrue ypred

10

ytrue 1 2 3

ypred 2 2 2

r2scoreytrue ypred

00

ytrue 1 2 3

ypred 3 2 1

r2scoreytrue ypred

30

Examples using sklearnmetricsr2score

•Effect of transforming the targets in regression model

•Linear Regression Example

•Lasso and Elastic Net for Sparse Signals

6244 Multilabel ranking metrics

See the Multilabel ranking metrics section of the user guide for further details

metricscoverageerror ytrue yscore Coverage error measure

metricslabelrankingaverageprecisionscore Compute rankingbased average precision

metricslabelrankingloss ytrue yscore Compute Ranking loss measure

sklearnmetrics coverageerror

sklearnmetrics coverageerror ytrue yscore sampleweightNone

Coverage error measure

Compute how far we need to go through the ranked scores to cover all true labels The best value is equal to the

average number of labels in ytrue per sample

Ties inyscores are broken by giving maximal rank that would have been assigned to all tied values

Note Our implementation’s score is 1 greater than the one given in Tsoumakas et al 2010 This extends it to

handle the degenerate case in which an instance has 0 true labels

Read more in the User Guide

Parameters

ytrue array shape nsamples nlabels True binary labels in binary indicator format

yscore array shape nsamples nlabels Target scores can either be probability esti

mates of the positive class confidence values or nonthresholded measure of decisions as

returned by “decisionfunction” on some classifiers

sampleweight arraylike of shape nsamples optional Sample weights

Returns

624sklearnmetrics Metrics 2023

scikitlearn user guide Release 0213

coverageerror float

References

1

sklearnmetrics labelrankingaverageprecisionscore

sklearnmetrics labelrankingaverageprecisionscore ytrue yscore sampleweightNone

Compute rankingbased average precision

Label ranking average precision LRAP is the average over each ground truth label assigned to each sample of the ratio of true vs total labels with lower score

This metric is used in multilabel ranking problem where the goal is to give better rank to the labels associated to each sample

The obtained score is always strictly greater than 0 and the best value is 1

Read more in the User Guide

Parameters

ytrue array or sparse matrix shape nsamples nlabels True binary labels in binary indicator format

yscore array shape nsamples nlabels Target scores can either be probability estimates of the positive class confidence values or nonthresholded measure of decisions as returned by “decisionfunction” on some classifiers

sampleweight arraylike of shape nsamples optional Sample weights

Returns

score float

Examples

```
import numpy as np
from sklearnmetrics import labelrankingaverageprecisionscore
```

```
ytrue nparray1 0 0 0 0 1
```

```
yscore nparray0.75 0.5 1 1 0.2 0.1
```

```
labelrankingaverageprecisionscore(ytrue, yscore)
```

0.416

sklearnmetrics labelrankingloss

sklearnmetrics labelrankingloss ytrue yscore sampleweightNone

Compute Ranking loss measure

Compute the average number of label pairs that are incorrectly ordered given yscore weighted by the size of the label set and the number of labels not in the label set

This is similar to the error set size but weighted by the number of relevant and irrelevant labels The best performance is achieved with a ranking loss of zero

2024 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Read more in the User Guide

New in version 017 A function labelrankingloss

Parameters

ytrue array or sparse matrix shape nsamples nlabels True binary labels in binary indicator format

yscore array shape nsamples nlabels Target scores can either be probability estimates of the positive class confidence values or nonthresholded measure of decisions as returned by “decisionfunction” on some classifiers

sampleweight arraylike of shape nsamples optional Sample weights

Returns

loss float

References

1

6245 Clustering metrics

See the Clustering performance evaluation section of the user guide for further details The sklearnmetrics cluster submodule contains evaluation metrics for cluster analysis results There are two forms of evaluation

- supervised which uses a ground truth class values for each sample
- unsupervised which does not and measures the ‘quality’ of the model itself

metricsadjustedmutualinfoscore

Adjusted Mutual Information between two clusterings

metricsadjustedrandscore labelstrue Rand index adjusted for chance

metricscalinskiharabaszscore X labels Compute the Calinski and Harabasz score

metricsdaviesbouldinscore X labels Computes the DaviesBouldin score

metricscompletenessscore labelstrue Completeness metric of a cluster labeling given a ground truth

metricsclustercontingencymatrix

Build a contingency matrix describing the relationship between labels

metricsfowlkesmallowsscore labelstrue

Measure the similarity of two clusterings of a set of points

metricshomogeneitycompletenessvmeasure Compute the homogeneity and completeness and V Measure scores at once

metricshomogeneityscore labelstrue Homogeneity metric of a cluster labeling given a ground truth

metricsmutualinfoscore labelstrue Mutual Information between two clusterings

metricsnormalizedmutualinfoscore

Normalized Mutual Information between two clusterings

metricssilhouettescore X labels Compute the mean Silhouette Coefficient of all samples

metricsilhouettesamples X labels metric Compute the Silhouette Coefficient for each sample

metricsvmeasurescore labelstrue la

belspredVmeasure cluster labeling given a ground truth

624sklearnmetrics Metrics 2025

scikitlearn user guide Release 0213

sklearnmetrics adjustedmutualinfoscore

sklearnmetrics adjustedmutualinfoscore labelstrue labelspred aver
agemethod'warn'

Adjusted Mutual Information between two clusterings

Adjusted Mutual Information AMI is an adjustment of the Mutual Information MI score to account for chance It accounts for the fact that the MI is generally higher for two clusterings with a larger number of clusters regardless of whether there is actually more information shared For two clusterings \mathcal{C}_1 and \mathcal{C}_2 the AMI is given as

$$AMI(\mathcal{C}_1, \mathcal{C}_2) = \frac{MI(\mathcal{C}_1, \mathcal{C}_2) - \frac{1}{2}MI(\mathcal{C}_1, \mathcal{C}_1) - \frac{1}{2}MI(\mathcal{C}_2, \mathcal{C}_2)}{1 - \frac{1}{2}MI(\mathcal{C}_1, \mathcal{C}_1) - \frac{1}{2}MI(\mathcal{C}_2, \mathcal{C}_2)}$$

This metric is independent of the absolute values of the labels a permutation of the class or cluster label values won't change the score value in any way

This metric is furthermore symmetric switching labeltrue withlabelpred will return the same score value This can be useful to measure the agreement of two independent label assignments strategies on the same dataset when the real ground truth is not known

Be mindful that this function is an order of magnitude slower than other metrics such as the Adjusted Rand Index

Read more in the User Guide

Parameters

labelstrue int array shape nsamples A clustering of the data into disjoint subsets

labelspred array shape nsamples A clustering of the data into disjoint subsets

averagemethod string optional default 'warn' How to compute the normalizer in the denominator Possible options are 'min' 'geometric' 'arithmetic' and 'max' If 'warn' 'max' will be used The default will change to 'arithmetic' in version 022

New in version 020

Returns

ami float upperlimited by 10 The AMI returns a value of 1 when the two partitions are identical ie perfectly matched Random partitions independent labellings have an expected AMI around 0 on average hence can be negative

See also

adjustedrandscore Adjusted Rand Index

mutualinfoscore Mutual Information not adjusted for chance

References

12

Examples

Perfect labelings are both homogeneous and complete hence have score 10

2026 Chapter 6 API (Application Programming Interface) Reference


```
scikitlearn user guide Release 0213
from sklearnmetricscluster import adjustedmutualinfoscore
adjustedmutualinfoscore0 0 1 1 0 0 1 1
```

```
10
adjustedmutualinfoscore0 0 1 1 1 1 0 0
```

```
10
If classes members are completely split across different clusters the assignment is totally incomplete hence
the AMI is null
adjustedmutualinfoscore0 0 0 0 0 1 2 3
```

```
00
Examples using sklearnmetricsadjustedmutualinfoscore
•Demo of affinity propagation clustering algorithm
•Demo of DBSCAN clustering algorithm
•Adjustment for chance in clustering performance evaluation
•A demo of KMeans clustering on the handwritten digits data
```

```
sklearnmetrics adjustedrandscore
sklearnmetrics adjustedrandscore labelstrue labelspred
Rand index adjusted for chance
```

The Rand Index computes a similarity measure between two clusterings by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings

The raw RI score is then “adjusted for chance” into the ARI score using the following scheme

```
ARI RI ExpectedRI maxRI ExpectedRI
The adjusted Rand index is thus ensured to have a value close to 00 for random labeling independently of the
number of clusters and samples and exactly 10 when the clusterings are identical up to a permutation
```

```
ARI is a symmetric measure
adjustedrandscorea b adjustedrandscoreb a
```

```
Read more in the User Guide
Parameters
labelstrue int array shape nsamples Ground truth class labels to be used as a reference
labelspred array shape nsamples Cluster labels to evaluate
```

```
Returns
arifloat Similarity score between 10 and 10 Random labelings have an ARI close to 00
10 stands for perfect match
See also
624sklearnmetrics Metrics 2027
```

scikitlearn user guide Release 0213
adjustedmutualinfoscore Adjusted Mutual Information
References
Hubert1985 wk
Examples
Perfectly matching labelings have a score of 1 even
from sklearnmetricscluster import adjustedrandscore
adjustedrandscore0 0 1 1 0 0 1 1
10
adjustedrandscore0 0 1 1 1 1 0 0
10
Labelings that assign all classes members to the same clusters are complete be not always pure hence penalized
adjustedrandscore0 0 1 2 0 0 1 1
057
ARI is symmetric so labelings that have pure clusters with members coming from the same classes but unnec
essary splits are penalized
adjustedrandscore0 0 1 1 0 0 1 2
057
If classes members are completely split across different clusters the assignment is totally incomplete hence the
ARI is very low
adjustedrandscore0 0 0 0 0 1 2 3
00
Examples using sklearnmetricsadjustedrandscore
•Demo of affinity propagation clustering algorithm
•Demo of DBSCAN clustering algorithm
•Adjustment for chance in clustering performance evaluation
•A demo of KMeans clustering on the handwritten digits data
•Clustering text documents using kmeans
sklearnmetrics calinskiharabaszscore
sklearnmetrics calinskiharabaszscore Xlabels
Compute the Calinski and Harabasz score
It is also known as the Variance Ratio Criterion
The score is defined as ratio between the withincluster dispersion and the betweencluster dispersion
Read more in the User Guide
2028 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures List of nfeatures dimensional data

points Each row corresponds to a single data point

labels arraylike shape nsamples Predicted labels for each sample

Returns

score float The resulting CalinskiHarabasz score

References

1

sklearnmetrics daviesbouldinscore

sklearnmetrics daviesbouldinscore Xlabels

Computes the DaviesBouldin score

The score is defined as the average similarity measure of each cluster with its most similar cluster where similarity is the ratio of withincluster distances to betweencluster distances Thus clusters which are farther apart and less dispersed will result in a better score

The minimum score is zero with lower values indicating better clustering

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures List of nfeatures dimensional data

points Each row corresponds to a single data point

labels arraylike shape nsamples Predicted labels for each sample

Returns

score float The resulting DaviesBouldin score

References

1

sklearnmetrics completenessscore

sklearnmetrics completenessscore labelstrue labelspred

Completeness metric of a cluster labeling given a ground truth

A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster

This metric is independent of the absolute values of the labels a permutation of the class or cluster label values won't change the score value in any way

This metric is not symmetric switching labeltrue withlabelpred will return the homogeneityscore which will be different in general

Read more in the User Guide

624sklearnmetrics Metrics 2029

scikitlearn user guide Release 0.21.3

Parameters

label_true int array shape (n_samples) ground truth class labels to be used as a reference

label_pred int array shape (n_samples) cluster labels to evaluate

Returns

completeness float score between 0.0 and 1.0 1.0 stands for perfectly complete labeling

See also

homogeneity_score

v_measure_score

References

1

Examples

Perfect labelings are complete

```
from sklearn.metrics.cluster import completeness_score
```

```
completeness_score([0, 1, 1, 1, 1, 0, 0]
```

```
[0, 1, 1, 1, 1, 0, 0]
```

Nonperfect labelings that assign all classes members to the same clusters are still complete

```
print completeness_score([0, 0, 1, 1, 0, 0, 0]
```

```
[0, 0, 1, 1, 0, 0, 0]
```

```
print completeness_score([0, 1, 2, 3, 0, 0, 1]
```

```
[0, 1, 2, 3, 0, 0, 1]
```

If classes members are split across different clusters the assignment cannot be complete

```
print completeness_score([0, 0, 1, 1, 0, 1, 0]
```

```
[0, 0, 1, 1, 0, 1, 0]
```

```
print completeness_score([0, 0, 0, 0, 0, 1, 2]
```

```
[0, 0, 0, 0, 0, 1, 2]
```

Examples using sklearn.metrics.completeness_score

- Demo of affinity propagation clustering algorithm

- Demo of DBSCAN clustering algorithm

- A demo of KMeans clustering on the handwritten digits data

- Clustering text documents using kmeans

2030 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearnmetricscluster contingencymatrix

sklearnmetricscluster contingencymatrix labelstrue labelspred epsNone
sparseFalse

Build a contingency matrix describing the relationship between labels

Parameters

labelstrue int array shape nsamples Ground truth class labels to be used as a reference

labelspred array shape nsamples Cluster labels to evaluate

eps None or float optional If a float that value is added to all values in the contingency

matrix This helps to stop NaN propagation If None nothing is adjusted

sparse boolean optional If True return a sparse CSR (Compressed Sparse Row) continency matrix If eps is not
None andsparse is True will throw ValueError

New in version 018

Returns

contingency arraylike sparse shapenclassestrue nclassespred Matrix such that

is the number of samples in true class and in predicted class Ifeps is None

the dtype of this array will be integer If eps is given the dtype will be float Will be a

scipysparsecsrmatrix ifsparseTrue

sklearnmetrics fowlkesmallowsscore

sklearnmetrics fowlkesmallowsscore labelstrue labelspred sparseFalse

Measure the similarity of two clusterings of a set of points

The FowlkesMallows index FMI is defined as the geometric mean between of the precision and recall

$FMI = \frac{TP}{\sqrt{TP + FP}} \cdot \frac{TP}{\sqrt{TP + FN}}$

WhereTPis the number of True Positive ie the number of pair of points that belongs in the same clusters

in bothlabelstrue andlabelspred FPis the number of False Positive ie the number of pair of

points that belongs in the same clusters in labelstrue and not inlabelspred andFNis the number

ofFalse Negative ie the number of pair of points that belongs in the same clusters in labelspred and not
inlabelsTrue

The score ranges from 0 to 1 A high value indicates a good similarity between two clusters

Read more in the User Guide

Parameters

labelstrue int array shape nsamples A clustering of the data into disjoint subsets

labelspred array shape nsamples A clustering of the data into disjoint subsets

sparse bool Compute contingency matrix internally with sparse matrix

Returns

score float The resulting FowlkesMallows score

References

12
624sklearnmetrics Metrics 2031

scikitlearn user guide Release 0213

Examples

Perfect labelings are both homogeneous and complete hence have score 10

from sklearnmetricscluster import fowlkesmallowsscore

fowlkesmallowsscore0 0 1 1 0 0 1 1

10

fowlkesmallowsscore0 0 1 1 1 1 0 0

10

If classes members are completely split across different clusters the assignment is totally random hence the FMI is null

fowlkesmallowsscore0 0 0 0 0 1 2 3

00

sklearnmetrics homogeneitycompletenessvmeasure

sklearnmetrics homogeneitycompletenessvmeasure labelstrue labelspred

beta10

Compute the homogeneity and completeness and VMeasure scores at once

Those metrics are based on normalized conditional entropy measures of the clustering labeling to evaluate given the knowledge of a Ground Truth class labels of the same samples

A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class

A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster

Both scores have positive values between 00 and 10 larger values being desirable

Those 3 metrics are independent of the absolute values of the labels a permutation of the class or cluster label values won't change the score values in any way

VMeasure is furthermore symmetric swapping labelstrue andlabelpred will give the same score This does not hold for homogeneity and completeness VMeasure is identical to normalizedmutualinfoscore with the arithmetic averaging method

Read more in the User Guide

Parameters

labelstrue int array shape nsamples ground truth class labels to be used as a reference

labelspred array shape nsamples cluster labels to evaluate

beta float Ratio of weight attributed to homogeneity vscompleteness Ifbeta is

greater than 1 completeness is weighted more strongly in the calculation If beta is less than 1 homogeneity is weighted more strongly

Returns

homogeneity float score between 00 and 10 10 stands for perfectly homogeneous labeling

completeness float score between 00 and 10 10 stands for perfectly complete labeling

vmeasure float harmonic mean of the first two

See also

2032 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

homogeneityscore

completenessscore

vmeasurescore

sklearnmetrics homogeneityscore

sklearnmetrics homogeneityscore labelstrue labelspred

Homogeneity metric of a cluster labeling given a ground truth

A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class

This metric is independent of the absolute values of the labels a permutation of the class or cluster label values won't change the score value in any way

This metric is not symmetric switching labeltrue withlabelpred will return the completenessscore which will be different in general

Read more in the User Guide

Parameters

labelstrue int array shape nsamples ground truth class labels to be used as a reference

labelspred array shape nsamples cluster labels to evaluate

Returns

homogeneity float score between 00 and 10 10 stands for perfectly homogeneous labeling

See also

completenessscore

vmeasurescore

References

1

Examples

Perfect labelings are homogeneous

from sklearnmetricscluster import homogeneityscore

homogeneityscore0 0 1 1 1 1 0 0

10

Nonperfect labelings that further split classes into more clusters can be perfectly homogeneous

print6f homogeneityscore0 0 1 1 0 0 1 2

1000000

print6f homogeneityscore0 0 1 1 0 1 2 3

1000000

624sklearnmetrics Metrics 2033

scikitlearn user guide Release 0213
Clusters that include samples from different classes do not make for an homogeneous labeling
print6f homogeneous_score0 0 1 1 0 1 0 1

00
print6f homogeneous_score0 0 1 1 0 0 0 0

00
Examples using sklearnmetrics.homogeneity_score
•Demo of affinity propagation clustering algorithm
•Demo of DBSCAN clustering algorithm
•A demo of KMeans clustering on the handwritten digits data
•Clustering text documents using kmeans

sklearnmetrics.mutual_info_score
sklearnmetrics.mutual_info_score(label_true, label_pred, contingency=None)
Mutual Information between two clusterings
The Mutual Information is a measure of the similarity between two labels of the same data. Where n_i is the number of the samples in cluster i and m_j is the number of the samples in cluster j , the Mutual Information between clusterings \mathcal{A} and \mathcal{B} is given as

$$MI(\mathcal{A}, \mathcal{B}) = \frac{1}{n} \sum_{i,j} \log \frac{n_{ij} n}{n_i n_j}$$

This metric is independent of the absolute values of the labels: a permutation of the class or cluster label values won't change the score value in any way.

This metric is furthermore symmetric: switching label_true with label_pred will return the same score value. This can be useful to measure the agreement of two independent label assignments strategies on the same dataset when the real ground truth is not known.

Read more in the User Guide

Parameters
label_true: int array shape (n_samples,) A clustering of the data into disjoint subsets
label_pred: array shape (n_samples,) A clustering of the data into disjoint subsets
contingency: None array sparse matrix shape (n_classes_true, n_classes_pred) A contingency matrix given by the contingency_matrix function. If value is None, it will be computed otherwise the given value is used with label_true and label_pred.
ignored

Returns
mi: float Mutual information, a nonnegative value

See also
adjusted_mutual_info_score Adjusted against chance Mutual Information
normalized_mutual_info_score Normalized Mutual Information
2034 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples using sklearnmetricsmutualinfoscore

•Adjustment for chance in clustering performance evaluation

sklearnmetrics normalizedmutualinfoscore

sklearnmetrics normalizedmutualinfoscore labelstrue labelspred averagemethod'warn'

Normalized Mutual Information between two clusterings

Normalized Mutual Information NMI is a normalization of the Mutual Information MI score to scale the results between 0 no mutual information and 1 perfect correlation In this function mutual information is normalized by some generalized mean of Hlabelstrue andHlabelspred defined by the averagemethod

This measure is not adjusted for chance Therefore adjustedmutualinfoscore might be preferred

This metric is independent of the absolute values of the labels a permutation of the class or cluster label values won't change the score value in any way

This metric is furthermore symmetric switching labeltrue withlabelpred will return the same score value This can be useful to measure the agreement of two independent label assignments strategies on the same dataset when the real ground truth is not known

Read more in the User Guide

Parameters

labelstrue int array shape nsamples A clustering of the data into disjoint subsets

labelspred array shape nsamples A clustering of the data into disjoint subsets

averagemethod string optional default 'warn' How to compute the normalizer in the denominator Possible options are 'min' 'geometric' 'arithmetic' and 'max' If 'warn' 'geometric' will be used The default will change to 'arithmetic' in version 022

New in version 020

Returns

nmi float score between 00 and 10 10 stands for perfectly complete labeling

See also

vmeasurescore VMeasure NMI with arithmetic mean option

adjustedrandscore Adjusted Rand Index

adjustedmutualinfoscore Adjusted Mutual Information adjusted against chance

Examples

Perfect labelings are both homogeneous and complete hence have score 10

from sklearnmetricscluster import normalizedmutualinfoscore

normalizedmutualinfoscore0 0 1 1 0 0 1 1

10

normalizedmutualinfoscore0 0 1 1 1 1 0 0

624sklearnmetrics Metrics 2035

10
If classes members are completely split across different clusters the assignment is totally incomplete hence the NMI is null
normalizedmutualinfoscore0 0 0 0 0 1 2 3

00
sklearnmetrics silhouette_score
sklearnmetrics silhouette_score Xlabels metric'euclidean' samplesizeNone ran
domstateNone kwds
Compute the mean Silhouette Coefficient of all samples
The Silhouette Coefficient is calculated using the mean intracluster distance a and the mean nearestcluster distance b for each sample The Silhouette Coefficient for a sample is $b - a / \max(a, b)$ To clarify b is the distance between a sample and the nearest cluster that the sample is not a part of Note that Silhouette Coefficient is only defined if number of labels is $2 \leq n_{\text{labels}} \leq n_{\text{samples}}$
This function returns the mean Silhouette Coefficient over all samples To obtain the values for each sample use `silhouette_samples`
The best value is 1 and the worst value is -1 Values near 0 indicate overlapping clusters Negative values generally indicate that a sample has been assigned to the wrong cluster as a different cluster is more similar
Read more in the User Guide

Parameters
Xarray `nsamples`a if `metric` "precomputed" or `nsamples`a
`nfeatures` otherwise Array of pairwise distances between samples or a feature array
`labels` array shape `(n_samples)` Predicted labels for each sample
`metric` string or callable The metric to use when calculating distance between instances in a feature array If `metric` is a string it must be one of the options allowed by `metrics.pairwise_distances` If `X` is the distance array itself use `metric=precomputed`
`samplesize` int or None The size of the sample to use when computing the Silhouette Coefficient on a random subset of the data If `samplesize` is None no sampling is used
`randomstate` int RandomState instance or None optional defaultNone The generator used to randomly select a subset of samples If int `randomstate` is the seed used by the random number generator If RandomState instance `randomstate` is the random number generator If None the random number generator is the RandomState instance used by `np.random` Used when `samplesize` is not None
`kwds` optional keyword parameters Any further parameters are passed directly to the distance function If using a `scipy.spatial.distance` metric the parameters are still metric dependent See the `scipy` docs for usage examples
Returns
`silhouette` float Mean Silhouette Coefficient for all samples
2036 Chapter 6 API (Application Programming Interface) Reference

References

12

Examples using sklearnmetrics.silhouettescore

- Demo of affinity propagation clustering algorithm
- Demo of DBSCAN clustering algorithm
- A demo of KMeans clustering on the handwritten digits data
- Selecting the number of clusters with silhouette analysis on KMeans clustering
- Clustering text documents using kmeans

sklearnmetrics.silhouettesamples

sklearnmetrics.silhouettesamples(Xlabels, metric='euclidean', kwds

Compute the Silhouette Coefficient for each sample

The Silhouette Coefficient is a measure of how well samples are clustered with samples that are similar to themselves. Clustering models with a high Silhouette Coefficient are said to be dense where samples in the same cluster are similar to each other and well separated where samples in different clusters are not very similar to each other.

The Silhouette Coefficient is calculated using the mean intracluster distance a and the mean nearestcluster distance b for each sample. The Silhouette Coefficient for a sample is $b - a / \max(a, b)$. Note that

Silhouette Coefficient is only defined if number of labels is $2 \leq n_{\text{labels}} \leq n_{\text{samples}} - 1$.

This function returns the Silhouette Coefficient for each sample.

The best value is 1 and the worst value is -1. Values near 0 indicate overlapping clusters.

Read more in the User Guide.

Parameters

X: array of shape (n_samples, n_features) or (n_samples, n_features) if metric="precomputed" or n_samples

n_features: otherwise Array of pairwise distances between samples or a feature array

labels: array of shape (n_samples,) label values for each sample

metric: string or callable. The metric to use when calculating distance between instances in

a feature array. If metric is a string it must be one of the options allowed by sklearn

metrics.pairwise_distances. If X is the distance array itself use

"precomputed" as the metric.

'kwds': optional keyword parameters. Any further parameters are passed directly to the dis

tance function. If using a scipy.spatial.distance metric the parameters are still

metric dependent. See the scipy docs for usage examples.

Returns

silhouette: array of shape (n_samples,) Silhouette Coefficient for each sample.

References

12

scikitlearn user guide Release 0213

Examples using sklearnmetricssilhouettesamples

•Selecting the number of clusters with silhouette analysis on KMeans clustering

sklearnmetrics vmeasurescore

sklearnmetrics vmeasurescore labelstrue labelspred beta10

Vmeasure cluster labeling given a ground truth

This score is identical to normalizedmutualinfoscore with thearithmetic option for averaging

The Vmeasure is the harmonic mean between homogeneity and completeness

$v = \frac{2}{\frac{1}{\beta} + 1}$ β homogeneity completeness

β homogeneity completeness

This metric is independent of the absolute values of the labels a permutation of the class or cluster label values won't change the score value in any way

This metric is furthermore symmetric switching labelstrue withlabelpred will return the same score

value This can be useful to measure the agreement of two independent label assignments strategies on the same dataset when the real ground truth is not known

Read more in the User Guide

Parameters

labelstrue int array shape nsamples ground truth class labels to be used as a reference

labelspred array shape nsamples cluster labels to evaluate

beta float Ratio of weight attributed to homogeneity vscompleteness Ifbeta is

greater than 1 completeness is weighted more strongly in the calculation If beta is

less than 1 homogeneity is weighted more strongly

Returns

vmeasure float score between 00 and 10 10 stands for perfectly complete labeling

See also

homogeneityscore

completenessscore

normalizedmutualinfoscore

References

1

Examples

Perfect labelings are both homogeneous and complete hence have score 10

2038 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

from sklearnmetricscluster import vmeasurescore

vmeasurescore0 0 1 1 0 0 1 1

10

vmeasurescore0 0 1 1 1 1 0 0

10

Labelings that assign all classes members to the same clusters are complete be not homogeneous hence penalized

print6f vmeasurescore0 0 1 2 0 0 1 1

08

print6f vmeasurescore0 1 2 3 0 0 1 1

066

Labelings that have pure clusters with members coming from the same classes are homogeneous but unnecessary splits harms completeness and thus penalize Vmeasure as well

print6f vmeasurescore0 0 1 1 0 0 1 2

08

print6f vmeasurescore0 0 1 1 0 1 2 3

066

If classes members are completely split across different clusters the assignment is totally incomplete hence the VMeasure is null

print6f vmeasurescore0 0 0 0 0 1 2 3

00

Clusters that include samples from totally different classes totally destroy the homogeneity of the labeling hence

print6f vmeasurescore0 0 1 1 0 0 0 0

00

Examples using sklearnmetricsvmeasurescore

- Biclustering documents with the Spectral Coclustering algorithm
- Demo of affinity propagation clustering algorithm
- Demo of DBSCAN clustering algorithm
- Adjustment for chance in clustering performance evaluation
- A demo of KMeans clustering on the handwritten digits data
- Clustering text documents using kmeans

6246 Biclustering metrics

See the Biclustering evaluation section of the user guide for further details

624sklearnmetrics Metrics 2039

scikitlearn user guide Release 0213

metricsconsensusscore a b similarity The similarity of two sets of biclusters

sklearnmetrics consensusscore

sklearnmetrics consensusscore absimilarity'jaccard'

The similarity of two sets of biclusters

Similarity between individual biclusters is computed Then the best matching between sets is found using the Hungarian algorithm The final score is the sum of similarities divided by the size of the larger set

Read more in the User Guide

Parameters

arows columns Tuple of row and column indicators for a set of biclusters

brows columns Another set of biclusters like a

similarity string or function optional default "jaccard" May be the string "jaccard" to use the Jaccard coefficient or any function that takes four arguments each of which is a 1d

indicator vector arows acolunns brows bcolumns

References

• Hochreiter Bodenhofer et al 2010 FABIA factor analysis for bicluster acquisition

Examples using sklearnmetricsconsensusscore

•A demo of the Spectral CoClustering algorithm

•A demo of the Spectral Biclustering algorithm

6247 Pairwise metrics

See the Pairwise metrics Affinities and Kernels section of the user guide for further details

metricspairwiseadditivechi2kernel X

YComputes the additive chisquared kernel between obser
vations in X and Y

metricspairwisechi2kernel X Y gamma Computes the exponential chisquared kernel X and Y

metricspairwisecosinesimilarity X Y

Compute cosine similarity between samples in X and Y

metricspairwisecosinedistances X Y Compute cosine distance between samples in X and Y

metricspairwisedistancemetrics Valid metrics for pairwisedistances

metricspairwiseeuclidean X

Y Considering the rows of X and YX as vectors compute

the distance matrix between each pair of vectors

metricspairwisehaversinedistances X

YCompute the Haversine distance between samples in X and
Y

metricspairwisekernelmetrics Valid metrics for pairwise kernels

metricspairwiselaplaciankernel X Y

gammaCompute the laplacian kernel between X and Y

metricspairwiselinearkernel X Y Compute the linear kernel between X and Y

Continued on next page

2040 Chapter 6 API (Application Programming Interface) Reference

metricspairwisemanhattandistances X

Y Compute the L1 distances between the vectors in X and Y

metricspairwisepairwise kernels X Y

Compute the kernel between arrays X and optional array Y

metricspairwisepolynomialkernel X Y

Compute the polynomial kernel between X and Y

metricspairwiserbfgkernel X Y gamma Compute the rbf gaussian kernel between X and Y

metricspairwis sigmoidkernel X Y Compute the sigmoid kernel between X and Y

metricspairwisepairedeuclidean distances X

Y Computes the paired euclidean distances between X and Y

metricspairwisepairedmanhattandistances X

Y Compute the L1 distances between the vectors in X and Y

metricspairwisepairedcosine distances X

Y Computes the paired cosine distances between X and Y

metricspairwisepaired distances X Y

metric Computes the paired distances between X and Y

metricspairwis edistances X Y metric Compute the distance matrix from a vector array X and op

tional Y

metricspairwis edistancesargmin X Y

Compute minimum distances between one point and a set

of points

metricspairwis edistancesargminmin X

Y Compute minimum distances between one point and a set

of points

metricspairwis edistanceschunked X Y

Generate a distance matrix chunk by chunk with optional

reduction

sklearnmetricspairwise additivechi2kernel

sklearnmetricspairwise additivechi2kernel X YNone

Computes the additive chisquared kernel between observations in X and Y

The chisquared kernel is computed between each pair of rows in X and Y X and Y have to be nonnegative

This kernel is most commonly applied to histograms

The chisquared kernel is given by

$$k_{x,y} = \sum_i \frac{x_i - y_i}{x_i + y_i}$$

It can be interpreted as a weighted difference per entry

Read more in the User Guide

Parameters

Xarraylike of shape nsamplesX nfeatures

Yarray of shape nsamplesY nfeatures

Returns

kernelmatrix array of shape nsamplesX nsamplesY

See also

chi2kernel The exponentiated version of the kernel which is usually preferable

sklearnkernelapproximationAdditiveChi2Sampler A Fourier approximation to this ker

nel

scikitlearn user guide Release 0213

Notes

As the negative of a distance this kernel is only conditionally positive definite

References

• Zhang J and Marszalek M and Lazebnik S and Schmid C Local features and kernels for classification of texture and object categories A comprehensive study International Journal of Computer Vision 2007 <https://research.microsoft.com/en-us/people/manik/projectstradeoffpapers/ZhangIJCV06.pdf>

`sklearn.metrics.pairwise.chi2kernel`

`sklearn.metrics.pairwise.chi2kernel(X, Y, None, gamma=10)`

Computes the exponential chisquared kernel X and Y

The chisquared kernel is computed between each pair of rows in X and Y X and Y have to be nonnegative

This kernel is most commonly applied to histograms

The chisquared kernel is given by

$$k(x, y) = \exp(\gamma \sum_i x_i^2 - x_i y_i)$$

It can be interpreted as a weighted difference per entry

Read more in the User Guide

Parameters

Xarraylike of shape `(n_samples, n_features)`

Yarray of shape `(n_samples, n_features)`

gamma float default=1 Scaling parameter of the chi2 kernel

Returns

kernelmatrix array of shape `(n_samples, n_samples)`

See also

`additivechi2kernel` The additive version of this kernel

`sklearn.kernel_approximation.AdditiveChi2Sampler` A Fourier approximation to the additive version of this kernel

References

• Zhang J and Marszalek M and Lazebnik S and Schmid C Local features and kernels for classification of texture and object categories A comprehensive study International Journal of Computer Vision 2007 <https://research.microsoft.com/en-us/people/manik/projectstradeoffpapers/ZhangIJCV06.pdf>

`sklearn.metrics.pairwise.cosinesimilarity`

`sklearn.metrics.pairwise.cosinesimilarity(X, Y, None, dense_output=True)`

Compute cosine similarity between samples in X and Y

Cosine similarity or the cosine kernel computes similarity as the normalized dot product of X and Y

2042 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

KX Y X Y XY

On L2normalized data this function is equivalent to linearkernel

Read more in the User Guide

Parameters

Xndarray or sparse array shape nsamplesX nfeatures Input data

Yndarray or sparse array shape nsamplesY nfeatures Input data If None the output

will be the pairwise similarities between all samples in X

denseoutput boolean optional default True Whether to return dense output even when the

input is sparse If False the output is sparse if both input arrays are sparse

New in version 017 parameter denseoutput for dense output

Returns

kernel matrix array An array with shape nsamplesX nsamplesY

sklearnmetricspairwise cosinedistances

sklearnmetricspairwise cosinedistances XYNone

Compute cosine distance between samples in X and Y

Cosine distance is defined as 10 minus the cosine similarity

Read more in the User Guide

Parameters

Xarraylike sparse matrix with shape nsamplesX nfeatures

Yarraylike sparse matrix optional with shape nsamplesY nfeatures

Returns

distance matrix array An array with shape nsamplesX nsamplesY

See also

sklearnmetricspairwisecosinesimilarity

scipyspatialdistancecosine dense matrices only

sklearnmetricspairwise distancemetrics

sklearnmetricspairwise distancemetrics

Valid metrics for pairwisedistances

This function simply returns the valid pairwise distance metrics It exists to allow for a description of the mapping for each of the valid strings

The valid distance metrics and the function they map to are

624sklearnmetrics Metrics 2043

scikitlearn user guide Release 0213

metric Function

'cityblock' metricspairwisemanhattandistances

'cosine' metricspairwisecosinedistances

'euclidean' metricspairwiseeuclidean

'haversine' metricspairwisehaversinedistances

'l1' metricspairwisemanhattandistances

'l2' metricspairwiseeuclidean

'manhattan' metricspairwisemanhattandistances

Read more in the User Guide

sklearnmetricspairwise euclidean

sklearnmetricspairwise euclidean X=None Y=None squared=False Xnormsquared=None

squared=False Xnormsquared=None

Considering the rows of X and Y as vectors compute the distance matrix between each pair of vectors

For efficiency reasons the euclidean distance between a pair of row vector x and y is computed as

$\sqrt{\sum (x_i - y_i)^2}$

This formulation has two advantages over other ways of computing distances First it is computationally efficient when dealing with sparse data Second if one argument varies but the other remains unchanged then

$\sum x_i^2$ and $\sum y_i^2$ can be precomputed

However this is not the most precise way of doing this computation and the distance matrix returned by this

function may not be exactly symmetric as required by eg scipyspatialdistance functions

Read more in the User Guide

Parameters

Xarraylike sparse matrix shape (n_samples1, n_features)

Yarraylike sparse matrix shape (n_samples2, n_features)

Ynormsquared arraylike shape (n_samples2,) optional Precomputed dotproducts of

vectors in Y eg $\sum y_i^2$ May be ignored in some cases see the note

below

squared boolean optional Return squared Euclidean distances

Xnormsquared arraylike shape (n_samples1,) optional Precomputed dotproducts of

vectors in X eg $\sum x_i^2$ May be ignored in some cases see the note

below

Returns

distances array shape (n_samples1, n_samples2)

See also

pairdistances distances between pairs of elements of X and Y

Notes

To achieve better accuracy Xnormsquared and Ynormsquared may be unused if they are passed as

float32

2044 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

from sklearnmetricspairwise import euclidean

X 0 1 1 1

distance between rows of X

euclidean

array0 1

1 0

get distance to origin

euclidean

array1

141421356

sklearnmetricspairwise haversine

sklearnmetricspairwise haversine XYNone

Compute the Haversine distance between samples in X and Y

The Haversine or great circle distance is the angular distance between two points on the surface of a sphere

The first distance of each point is assumed to be the latitude the second is the longitude given in radians The

dimension of the data must be 2

$$d = 2 \arcsin \sqrt{\sin^2 \frac{\phi_1 - \phi_2}{2} + \cos \phi_1 \cos \phi_2 \sin^2 \frac{\lambda_1 - \lambda_2}{2}}$$

Parameters

Xarraylike shape nsamples1 2

Yarraylike shape nsamples2 2 optional

Returns

distance array shape nsamples1 nsamples2

Notes

As the Earth is nearly spherical the haversine formula provides a good approximation of the distance between

two points of the Earth surface with a less than 1 error on average

Examples

We want to calculate the distance between the Ezeiza Airport Buenos Aires Argentina and the Charles de

Gaulle Airport Paris France

from sklearnmetricspairwise import haversine

bsas 3483333 585166646

paris 490083899664 253844117956

result haversinebsas paris

result63710001000 multiply by Earth radius to get kilometers

array0 1127945379464

1127945379464 0

624sklearnmetrics Metrics 2045

scikitlearn user guide Release 0213

sklearnmetricspairwise kernelmetrics

sklearnmetricspairwise kernelmetrics

Valid metrics for pairwise kernels

This function simply returns the valid pairwise distance metrics It exists however to allow for a verbose description of the mapping for each of the valid strings

The valid distance metrics and the function they map to are

metric Function

‘additivechi2’ sklearnpairwiseadditivechi2kernel

‘chi2’ sklearnpairwisechi2kernel

‘linear’ sklearnpairwiselinearkernel

‘poly’ sklearnpairwisepolynomialkernel

‘polynomial’ sklearnpairwisepolynomialkernel

‘rbf’ sklearnpairwiserbfkernel

‘laplacian’ sklearnpairwiselaplaciankernel

‘sigmoid’ sklearnpairwisesigmoidkernel

‘cosine’ sklearnpairwisecosinesimilarity

Read more in the User Guide

sklearnmetricspairwise laplaciankernel

sklearnmetricspairwise laplaciankernel XYNone gammaNone

Compute the laplacian kernel between X and Y

The laplacian kernel is defined as

$$K_{xy} = \exp(-\gamma \|x - y\|)$$

for each pair of rows x in X and y in Y Read more in the User Guide

New in version 017

Parameters

Xarray of shape nsamplesX nfeatures

Yarray of shape nsamplesY nfeatures

gamma float default None If None defaults to $10 / n_{\text{features}}$

Returns

kernelmatrix array of shape nsamplesX nsamplesY

sklearnmetricspairwise linearkernel

sklearnmetricspairwise linearkernel XYNone denseoutputTrue

Compute the linear kernel between X and Y

Read more in the User Guide

Parameters

Xarray of shape nsamples1 nfeatures

2046 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Yarray of shape nsamples2 nfeatures

denseoutput boolean optional default True Whether to return dense output even when the input is sparse If False the output is sparse if both input arrays are sparse

New in version 020

Returns

Gram matrix array of shape nsamples1 nsamples2

sklearnmetricspairwise manhattandistances

sklearnmetricspairwise manhattandistances XYNone sumoverfeaturesTrue

Compute the L1 distances between the vectors in X and Y

With sumoverfeatures equal to False it returns the componentwise distances

Read more in the User Guide

Parameters

Xarraylike An array with shape nsamplesX nfeatures

Yarraylike optional An array with shape nsamplesY nfeatures

sumoverfeatures bool defaultTrue If True the function returns the pairwise distance matrix else it returns the componentwise L1 pairwisedistances Not supported for sparse matrix inputs

Returns

Darray If sumoverfeatures is False shape is nsamplesX nsamplesY nfeatures and D contains the componentwise L1 pairwisedistances ie absolute difference else shape is nsamplesX nsamplesY and D contains the pairwise L1 distances

Examples

```
from sklearnmetricspairwise import manhattandistances
manhattandistances3 3
array0
manhattandistances3 2
array1
manhattandistances2 3
array1
manhattandistances1 2 3 4 1 2 0 3
array0 2
4 4
import numpy as np
X np.ones1 2
y np.full2 2 2
manhattandistancesX y sumoverfeatures False
array1 1
1 1
```

624sklearnmetrics Metrics 2047

scikitlearn user guide Release 0213

sklearnmetricspairwise pairwise kernels

sklearnmetricspairwise pairwise kernels X YNone metric'linear' fil

terparamsFalse njobsNone kwds

Compute the kernel between arrays X and optional array Y

This method takes either a vector array or a kernel matrix and returns a kernel matrix If the input is a vector array the kernels are computed If the input is a kernel matrix it is returned instead

This method provides a safe way to take a kernel matrix as input while preserving compatibility with many other algorithms that take a vector array

If Y is given default is None then the returned matrix is the pairwise kernel between the arrays from both X and Y

Valid values for metric are

'additivechi2' 'chi2' 'linear' 'poly' 'polynomial' 'rbf' 'laplacian' 'sigmoid' 'cosine'

Read more in the User Guide

Parameters

Xarray nsamplesa nsamplesa if metric "precomputed" or nsamplesa

nfeatures otherwise Array of pairwise kernels between samples or a feature array

Yarray nsamplesb nfeatures A second feature array only if X has shape nsamplesa

nfeatures

metric string or callable The metric to use when calculating kernel between instances

in a feature array If metric is a string it must be one of the metrics in pair

wisePAIRWISEKERNelfUNCTIONS If metric is "precomputed" X is assumed to

be a kernel matrix Alternatively if metric is a callable function it is called on each pair of instances rows and the resulting value recorded The callable should take two arrays from

X as input and return a value indicating the distance between them

filterparams boolean Whether to filter invalid parameters or not

njobs int or None optional defaultNone The number of jobs to use for the computation

This works by breaking down the pairwise matrix into njobs even slices and computing them in parallel

None means 1 unless in a joblibparallelbackend context1means using all

processors See Glossary for more details

kwds optional keyword parameters Any further parameters are passed directly to the kernel function

Returns

Karray nsamplesa nsamplesa or nsamplesa nsamplesb A kernel matrix K such

that K_{ij} is the kernel between the i th and j th vectors of the given matrix X if Y is None

If Y is not None then K_{ij} is the kernel between the i th array from X and the j th array from Y

Notes

If metric is 'precomputed' Y is ignored and X is returned

2048 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearnmetricspairwise polynomialkernel

sklearnmetricspairwise polynomialkernel XYNone degree3 gammaNone
coef01

Compute the polynomial kernel between X and Y

$K(X,Y) = \sum_{j=0}^{\text{degree}} \text{coef0} \gamma^j (X \cdot Y)^j$

Read more in the User Guide

Parameters

Xndarray of shape nsamples1 nfeatures

Yndarray of shape nsamples2 nfeatures

degree int default 3

gamma float default None if None defaults to $1/(n \text{ features} + 1)$

coef0 float default 1

Returns

Gram matrix array of shape nsamples1 nsamples2

sklearnmetricspairwise rbfkernel

sklearnmetricspairwise rbfkernel XYNone gammaNone

Compute the rbf gaussian kernel between X and Y

$K(x,y) = \exp(-\gamma \|x - y\|^2)$

for each pair of rows x in X and y in Y

Read more in the User Guide

Parameters

Xarray of shape nsamplesX nfeatures

Yarray of shape nsamplesY nfeatures

gamma float default None If None defaults to $1/(n \text{ features} + 1)$

Returns

kernelmatrix array of shape nsamplesX nsamplesY

sklearnmetricspairwise sigmoidkernel

sklearnmetricspairwise sigmoidkernel XYNone gammaNone coef01

Compute the sigmoid kernel between X and Y

$K(X,Y) = \frac{1}{1 + \exp(-\gamma X \cdot Y)}$

Read more in the User Guide

Parameters

Xndarray of shape nsamples1 nfeatures

Yndarray of shape nsamples2 nfeatures

624sklearnmetrics Metrics 2049

scikitlearn user guide Release 0213

gamma float default None If None defaults to 10 nfeatures

coef0 float default 1

Returns

Gram matrix array of shape nsamples1 nsamples2

sklearnmetricspairwise pairedeuclidean

sklearnmetricspairwise pairedeuclidean XY

Computes the paired euclidean distances between X and Y

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures

Yarraylike shape nsamples nfeatures

Returns

distances ndarray nsamples

sklearnmetricspairwise pairedmanhattan

sklearnmetricspairwise pairedmanhattan XY

Compute the L1 distances between the vectors in X and Y

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures

Yarraylike shape nsamples nfeatures

Returns

distances ndarray nsamples

sklearnmetricspairwise pairedcosine

sklearnmetricspairwise pairedcosine XY

Computes the paired cosine distances between X and Y

Read more in the User Guide

Parameters

Xarraylike shape nsamples nfeatures

Yarraylike shape nsamples nfeatures

Returns

distances ndarray shape nsamples

2050 Chapter 6 API (Application Programming Interface) Reference

Notes

The cosine distance is equivalent to the half the squared euclidean distance if each sample is normalized to unit norm

sklearnmetricspairwise paireddistances
sklearnmetricspairwise paireddistances XYmetric'euclidean' kwds

Computes the paired distances between X and Y
Computes the distances between X0 Y0 X1 Y1 etc

Read more in the User Guide

Parameters

Xndarray nsamples nfeatures Array 1 for distance computation
Yndarray nsamples nfeatures Array 2 for distance computation
metric string or callable The metric to use when calculating distance between instances in a feature array If metric is a string it must be one of the options specified in PAIREDDISTANCES including "euclidean" "manhattan" or "cosine" Alternatively if metric is a callable function it is called on each pair of instances rows and the resulting value recorded The callable should take two arrays from X as input and return a value indicating the distance between them

Returns

distances ndarray nsamples

See also

pairwisedistances Computes the distance between every pair of samples

Examples

from sklearnmetricspairwise import paireddistances

X 0 1 1 1

Y 0 1 2 1

paireddistancesX Y

array0 1

sklearnmetrics pairwisedistances

sklearnmetrics pairwisedistances XYNone metric'euclidean' njobsNone kwds

Compute the distance matrix from a vector array X and optional Y

This method takes either a vector array or a distance matrix and returns a distance matrix If the input is a vector array the distances are computed If the input is a distances matrix it is returned instead

This method provides a safe way to take a distance matrix as input while preserving compatibility with many other algorithms that take a vector array

If Y is given default is None then the returned matrix is the pairwise distance between the arrays from both X and Y

scikitlearn user guide Release 0213

Valid values for metric are

- From scikitlearn ‘cityblock’ ‘cosine’ ‘euclidean’ ‘l1’ ‘l2’ ‘manhattan’ These metrics support sparse matrix inputs
- From scipyspatialdistance ‘braycurtis’ ‘canberra’ ‘chebyshev’ ‘correlation’ ‘dice’ ‘hamming’ ‘jaccard’ ‘kulsinski’ ‘mahalanobis’ ‘minkowski’ ‘rogerstanimoto’ ‘russellrao’ ‘seuclidean’ ‘sokalmichener’ ‘sokalsneath’ ‘sqeuclidean’ ‘yule’ See the documentation for scipyspatialdistance for details on these metrics These metrics do not support sparse matrix inputs

Note that in the case of ‘cityblock’ ‘cosine’ and ‘euclidean’ which are valid scipyspatialdistance metrics the scikitlearn implementation will be used which is faster and has support for sparse matrices except for ‘cityblock’ For a verbose description of the metrics from scikitlearn see the doc of the sklearnpairwiselocalmetrics function

Read more in the User Guide

Parameters

Xarray nsamplesa nsamplesa if metric “precomputed” or nsamplesa

nfeatures otherwise Array of pairwise distances between samples or a feature array

Yarray nsamplesb nfeatures optional An optional second feature array Only allowed if metric “precomputed”

metric string or callable The metric to use when calculating distance between instances in a feature array If metric is a string it must be one of the options allowed by scipyspatialdistancepairwise for its metric parameter or a metric listed in pairwisePAIRWISEDISTANCEFUNCTIONS If metric is “precomputed” X is assumed to be a distance matrix Alternatively if metric is a callable function it is called on each pair of instances rows and the resulting value recorded The callable should take two arrays from X as input and return a value indicating the distance between them

njobs int or None optional defaultNone The number of jobs to use for the computation

This works by breaking down the pairwise matrix into njobs even slices and computing them in parallel

None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

kwds optional keyword parameters Any further parameters are passed directly to the distance function If using a scipyspatialdistance metric the parameters are still metric dependent See the scipy docs for usage examples

Returns

Darray nsamplesa nsamplesa or nsamplesa nsamplesb A distance matrix D

such that D_{ij} is the distance between the i th and j th vectors of the given matrix X if Y is None If Y is not None then D_{ij} is the distance between the i th array from X and the j th array from Y

See also
pairwiselocalmetrics performs the same calculation as this function but returns a generator of chunks of the distance matrix in order to limit memory usage
paireddistances Computes the distances between corresponding elements of two arrays
2052 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples using sklearnmetricspairwiseditances

•Agglomerative clustering with different metrics

sklearnmetrics pairwiseditancesargmin

sklearnmetrics pairwiseditancesargmin X Y axis1 metric'euclidean'

batchsizeNone metrickwargsNone

Compute minimum distances between one point and a set of points

This function computes for each row in X the index of the row of Y which is closest according to the specified distance

This is mostly equivalent to calling

pairwiseditancesX YY metricmetricargminaxisaxis

but uses much less memory and is faster for large arrays

This function works with dense 2D arrays only

Parameters

Xarraylike Arrays containing points Respective shapes nsamples1 nfeatures and

nsamples2 nfeatures

Yarraylike Arrays containing points Respective shapes nsamples1 nfeatures and

nsamples2 nfeatures

axis int optional default 1 Axis along which the argmin and distances are to be computed

metric string or callable metric to use for distance computation Any metric from scikitlearn

or scipyspatialdistance can be used

If metric is a callable function it is called on each pair of instances rows and the resulting

value recorded The callable should take two arrays as input and return one value indicating

the distance between them This works for Scipy's metrics but is less efficient than passing

the metric name as a string

Distance matrices are not supported

Valid values for metric are

• from scikitlearn 'cityblock' 'cosine' 'euclidean' 'l1' 'l2' 'manhattan'

• from scipyspatialdistance 'braycurtis' 'canberra' 'chebyshev' 'correlation' 'dice'

'hamming' 'jaccard' 'kulsinski' 'mahalanobis' 'minkowski' 'rogerstanimoto' 'rus

sellrao' 'seuclidean' 'sokalmichener' 'sokalsneath' 'sqeuclidean' 'yule'

See the documentation for scipyspatialdistance for details on these metrics

batchsize integer Deprecated since version 020 Deprecated for removal in 022 Use

sklearnsetconfigworkingmemory instead

metrickwargs dict keyword arguments to pass to specified metric function

Returns

argmin numpyndarray Yargmini is the row in Y that is closest to Xi

See also

sklearnmetricspairwiseditances

624sklearnmetrics Metrics 2053

scikitlearn user guide Release 0213

sklearnmetricspairwiselargminmin

Examples using sklearnmetricspairwiselargminmin

•Color Quantization using KMeans

•Comparison of the KMeans and MiniBatchKMeans clustering algorithms

sklearnmetrics pairwiselargminmin

sklearnmetrics pairwiselargminmin XYaxis1 metric'euclidean'

batchsizeNone metrickwargsNone

Compute minimum distances between one point and a set of points

This function computes for each row in X the index of the row of Y which is closest according to the specified distance The minimal distances are also returned

This is mostly equivalent to calling

pairwiselargminminX YY metricmetricargminaxisaxis pairwiselargminminX YY

metricmetricminaxisaxis

but uses much less memory and is faster for large arrays

Parameters

Xarraylike sparse matrix shape nsamples1 nfeatures Array containing points

Yarraylike sparse matrix shape nsamples2 nfeatures Arrays containing points

axis int optional default 1 Axis along which the argmin and distances are to be computed

metric string or callable default 'euclidean' metric to use for distance computation Any

metric from scikitlearn or scipyspatialdistance can be used

If metric is a callable function it is called on each pair of instances rows and the resulting value recorded The callable should take two arrays as input and return one value indicating the distance between them This works for Scipy's metrics but is less efficient than passing the metric name as a string

Distance matrices are not supported

Valid values for metric are

- from scikitlearn 'cityblock' 'cosine' 'euclidean' 'l1' 'l2' 'manhattan'
- from scipyspatialdistance 'braycurtis' 'canberra' 'chebyshev' 'correlation' 'dice' 'hamming' 'jaccard' 'kulsinski' 'mahalanobis' 'minkowski' 'rogerstanimoto' 'rus sellrao' 'seuclidean' 'sokalmichener' 'sokalsneath' 'sqeuclidean' 'yule'

See the documentation for scipyspatialdistance for details on these metrics

batchsize integer Deprecated since version 020 Deprecated for removal in 022 Use

sklearnsetconfigworkingmemory instead

metrickwargs dict optional Keyword arguments to pass to specified metric function

Returns

argmin numpyndarray Yargmini is the row in Y that is closest to Xi

distances numpyndarray distancesi is the distance between the ith row in X and the argminith row in Y

2054 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

See also

sklearnmetricspairwisedistances

sklearnmetricspairwisedistancesargmin

sklearnmetrics pairwisedistanceschunked

sklearnmetrics pairwisedistanceschunked XYNone reducefuncNone met

ric'euclidean' njobsNone work

ingmemoryNone kwds

Generate a distance matrix chunk by chunk with optional reduction

In cases where not all of a pairwise distance matrix needs to be stored at once this is used to calculate pairwise distances in workingmemory sized chunks If reducefunc is given it is run on each chunk and its

return values are concatenated into lists arrays or sparse matrices

Parameters

Xarray nsamplesa nsamplesa if metric "precomputed" or nsamplesa

nfeatures otherwise Array of pairwise distances between samples or a feature array

Yarray nsamplesb nfeatures optional An optional second feature array Only allowed

if metric "precomputed"

reducefunc callable optional The function which is applied on each chunk of the distance matrix reducing it to needed values reducefuncDchunk start is called re

peatedly where Dchunk is a contiguous vertical slice of the pairwise distance matrix

starting at row start It should return an array a list or a sparse matrix of length

Dchunkshape0 or a tuple of such objects

If None pairwisedistanceschunked returns a generator of vertical chunks of the distance matrix

metric string or callable The metric to use when calculating distance between in

stances in a feature array If metric is a string it must be one of the options al

lowed by scipyspatialdistancepdist for its metric parameter or a metric listed in pair

wisePAIRWISEDISTANCEFUNCTIONS If metric is "precomputed" X is assumed to

be a distance matrix Alternatively if metric is a callable function it is called on each pair

of instances rows and the resulting value recorded The callable should take two arrays

from X as input and return a value indicating the distance between them

njobs int or None optional defaultNone The number of jobs to use for the computation

This works by breaking down the pairwise matrix into njobs even slices and computing them in parallel

None means 1 unless in a joblibparallelbackend context1means using all

processors See Glossary for more details

workingmemory int optional The sought maximum memory for temporary

distance matrix chunks When None default the value of sklearn

getconfigworkingmemory is used

'kwds' optional keyword parameters Any further parameters are passed directly to the dis

tance function If using a scipyspatialdistance metric the parameters are still metric de

pendent See the scipy docs for usage examples

Yields

624sklearnmetrics Metrics 2055

scikitlearn user guide Release 0213

Dchunk array or sparse matrix A contiguous slice of distance matrix optionally processed

byreducefunc

Examples

Without reducefunc

```
import numpy as np
from sklearnmetrics import pairwisedistanceschunked
X np.random.RandomState(0).rand(5, 3)
```

Dchunk nextpairwisedistanceschunkedX

Dchunk

array0 0.29 0.41 0.19 0.57

0.29 0 0.57 0.41 0.76

0.41 0.57 0 0.44 0.90

0.19 0.41 0.44 0 0.51

0.57 0.76 0.90 0.51 0

Retrieve all neighbors and average distance within radius r

r = 2

```
def reducefunc(Dchunk, start, neigh, np.flatnonzero, d, r, for_in, Dchunk):
    avgdist = Dchunk[Dchunk, r].mean(axis=1)
    return neigh, avgdist

gen = pairwisedistanceschunked(X, reducefunc, reducefunc)
neigh, avgdist = next(gen)
neigh
array0 3 array1 array2 array0 3 array4
avgdist
array0 0.39 0 0 0.039 0
```

Where r is defined per sample we need to make use of start

r = [2, 4, 4, 3, 1]

```
def reducefunc(Dchunk, start):
    neigh = np.flatnonzero(d[r])
    for i, d in enumerate(Dchunk[start]):
        return neigh, nextpairwisedistanceschunked(X, reducefunc, reducefunc)
neigh, nextpairwisedistanceschunked(X, reducefunc, reducefunc)
neigh
array0 3 array0 1 array2 array0 3 array4
```

Force row-by-row generation by reducing working memory

```
gen = pairwisedistanceschunked(X, reducefunc, reducefunc,
                               workingmemory=0)
next(gen)
array0 3
next(gen)
array0 1
```

625sklearnmixture Gaussian Mixture Models

The sklearnmixture module implements mixture modeling algorithms

2056 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

User guide See the Gaussian mixture models section for further details

mixtureBayesianGaussianMixture Variational Bayesian estimation of a Gaussian mixture

mixtureGaussianMixture ncomponents Gaussian Mixture

6251sklearnmixture BayesianGaussianMixture

classsklearnmixture BayesianGaussianMixture ncomponents1 covariancetype'full'

tol0001 regcovar1e06 maxiter100

ninit1 initparams'kmeans'

weightconcentrationpriortype'dirichletprocess'

weightconcentrationpriorNone

meanprecisionpriorNone

meanpriorNone de

greessofffreedompriorNone covari

ancepriorNone randomstateNone

warmstartFalse verbose0 ver

boseinterval10

Variational Bayesian estimation of a Gaussian mixture

This class allows to infer an approximate posterior distribution over the parameters of a Gaussian mixture

distribution The effective number of components can be inferred from the data

This class implements two types of prior for the weights distribution a finite mixture model with Dirichlet

distribution and an infinite mixture model with the Dirichlet Process In practice Dirichlet Process inference

algorithm is approximated and uses a truncated distribution with a fixed maximum number of components

called the Stickbreaking representation The number of components actually used almost always depends on

the data

New in version 018

Read more in the User Guide

Parameters

ncomponents int defaults to 1 The number of mixture components Depending on the data

and the value of the weightconcentrationprior the model can decide to not use

all the components by setting some component weights to values very close to zero

The number of effective components is therefore smaller than ncomponents

covariancetype 'full' 'tied' 'diag' 'spherical' defaults to 'full' String describing the

type of covariance parameters to use Must be one of

full each component has its own general covariance matrix

tied all components share the same general covariance matrix

diag each component has its own diagonal covariance matrix

spherical each component has its own single variance

tolfloat defaults to 1e3 The convergence threshold EM iterations will stop when the lower

bound average gain on the likelihood of the training data with respect to the model is below

this threshold

regcovar float defaults to 1e6 Nonnegative regularization added to the diagonal of co

variance Allows to assure that the covariance matrices are all positive

maxiter int defaults to 100 The number of EM iterations to perform

625sklearnmixture Gaussian Mixture Models 2057

scikitlearn user guide Release 0213

ninit int defaults to 1 The number of initializations to perform The result with the highest lower bound value on the likelihood is kept

initparams ‘kmeans’ ‘random’ defaults to ‘kmeans’ The method used to initialize the weights the means and the covariances Must be one of

kmeans responsibilities are initialized using kmeans

random responsibilities are initialized randomly

weightconcentrationpriortype str defaults to ‘dirichletprocess’ String describing the type of the weight concentration prior Must be one of

dirichletprocess using the Stickbreaking representation

dirichletdistribution can favor more uniform weights

weightconcentrationprior float None optional The dirichlet concentration of each component on the weight distribution Dirichlet This is commonly called gamma in the literature The higher concentration puts more mass in the center and will lead to more components being active while a lower concentration parameter will lead to more mass at the edge of the mixture weights simplex The value of the parameter must be greater than 0 If it is None it’s set to 1 ncomponents

meanprecisionprior float None optional The precision prior on the mean distribution Gaussian Controls the extend to where means can be placed Larger values concentrate the means of each clusters around meanprior The value of the parameter must be greater than 0 If it is None it’s set to 1

meanprior arraylike shape nfeatures optional The prior on the mean distribution Gaussian If it is None it’s set to the mean of X

degreesoffreedomprior float None optional The prior of the number of degrees of freedom on the covariance distributions Wishart If it is None it’s set to nfeatures

covarianceprior float or arraylike optional The prior on the covariance distribution Wishart If it is None the emiprical covariance prior is initialized using the covariance of X The shape depends on covariancetype

nfeatures nfeatures iffull

nfeatures nfeatures iftied

nfeatures ifdiag

float ifspherical

randomstate int RandomState instance or None optional defaultNone If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

warmstart bool default to False If ‘warmstart’ is True the solution of the last fitting is used as initialization for the next call of fit This can speed up convergence when fit is called several times on similar problems See the Glossary

verbose int default to 0 Enable verbose output If 1 then it prints the current initialization and each iteration step If greater than 1 then it prints also the log probability and the time needed for each step

verboseinterval int default to 10 Number of iteration done before the next print

Attributes

weights arraylike shape ncomponents The weights of each mixture components

2058 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

means arraylike shape ncomponents nfeatures The mean of each mixture component
covariances arraylike The covariance of each mixture component The shape depends on
covariancetype

ncomponents ifspherical
nfeatures nfeatures iftied
ncomponents nfeatures ifdiag
ncomponents nfeatures nfeatures iffull

precisions arraylike The precision matrices for each component in the mixture A preci
sion matrix is the inverse of a covariance matrix A covariance matrix is symmetric posi
tive definite so the mixture of Gaussian can be equivalently parameterized by the precision
matrices Storing the precision matrices instead of the covariance matrices makes it more
efficient to compute the loglikelihood of new samples at test time The shape depends on
covariancetype

ncomponents ifspherical
nfeatures nfeatures iftied
ncomponents nfeatures ifdiag
ncomponents nfeatures nfeatures iffull

precisionscholesky arraylike The cholesky decomposition of the precision matrices of
each mixture component A precision matrix is the inverse of a covariance matrix A covari
ance matrix is symmetric positive definite so the mixture of Gaussian can be equivalently
parameterized by the precision matrices Storing the precision matrices instead of the co
variance matrices makes it more efficient to compute the loglikelihood of new samples at
test time The shape depends on covariancetype

ncomponents ifspherical
nfeatures nfeatures iftied
ncomponents nfeatures ifdiag
ncomponents nfeatures nfeatures iffull

converged bool True when convergence was reached in fit False otherwise
niter int Number of step used by the best fit of inference to reach the convergence
lowerbound float Lower bound value on the likelihood of the training data with respect to
the model of the best fit of inference

weightconcentrationprior tuple or float The dirichlet concentration of each
component on the weight distribution Dirichlet The type depends on
weightconcentrationpriortype

float float ifdirichletprocess Beta parameters
float ifdirichletdistribution Dirichlet parameters

The higher concentration puts more mass in the center and will lead to more components
being active while a lower concentration parameter will lead to more mass at the edge of
the simplex

weightconcentration arraylike shape ncomponents The dirichlet concentration of
each component on the weight distribution Dirichlet

meanprecisionprior float The precision prior on the mean distribution Gaussian Con
trols the extend to where means can be placed Larger values concentrate the means of each
clusters around meanprior

625sklearnmixture Gaussian Mixture Models 2059

scikitlearn user guide Release 0213

meanprecision arraylike shape ncomponents The precision of each components on the mean distribution Gaussian

meanprior arraylike shape nfeatures The prior on the mean distribution Gaussian

degreesoffreedomprior float The prior of the number of degrees of freedom on the covariance distributions Wishart

degreesoffreedom arraylike shape ncomponents The number of degrees of freedom of each components in the model

covarianceprior float or arraylike The prior on the covariance distribution Wishart The shape depends on covariancetype

nfeatures nfeatures iffull

nfeatures nfeatures iftied

nfeatures ifdiag

float ifspherical

See also

GaussianMixture Finite Gaussian mixture fit with EM

References

R16529824bff21 R16529824bff22 R16529824bff23

Methods

fitself X y Estimate model parameters with the EM algorithm

fitpredict self X y Estimate model parameters using X and predict the labels for X

getparams self deep Get parameters for this estimator

predict self X Predict the labels for the data samples in X using trained model

predictproba self X Predict posterior probability of each component given the data

sample self nsamples Generate random samples from the fitted Gaussian distribution

score self X y Compute the persample average loglikelihood of the given data X

scoresamples self X Compute the weighted log probabilities for each sample

setparams self params Set the parameters of this estimator

init selfncomponents1 covariancetype'full' tol0001 regcovar1e06 maxiter100

ninit1 initparams'kmeans' weightconcentrationpriortype'dirichletprocess'

weightconcentrationpriorNone meanprecisionpriorNone meanpriorNone

degreesoffreedompriorNone covariancepriorNone randomstateNone

warmstartFalse verbose0 verboseinterval10

fitselfXyNone

Estimate model parameters with the EM algorithm

2060 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The method fits the model ninit times and sets the parameters with which the model has the largest likelihood or lower bound Within each trial the method iterates between Estep and Mstep for maxiter times until the change of likelihood or lower bound is less than tol otherwise a ConvergenceWarning is raised If warmstart isTrue then ninit is ignored and a single initialization is performed upon the first call Upon consecutive calls training starts where it left off

Parameters

Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points

Each row corresponds to a single data point

Returns

self

fitpredict selfXyNone

Estimate model parameters using X and predict the labels for X

The method fits the model ninit times and sets the parameters with which the model has the largest likelihood or lower bound Within each trial the method iterates between Estep and Mstep for maxiter times until the change of likelihood or lower bound is less than tol otherwise a ConvergenceWarning is raised After fitting it predicts the most probable label for the input data points

New in version 020

Parameters

Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points

Each row corresponds to a single data point

Returns

labels array shape nsamples Component labels

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict the labels for the data samples in X using trained model

Parameters

Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points

Each row corresponds to a single data point

Returns

labels array shape nsamples Component labels

predictproba selfX

Predict posterior probability of each component given the data

Parameters

Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points

Each row corresponds to a single data point

625sklearnmixture Gaussian Mixture Models 2061

scikitlearn user guide Release 0213

Returns

resp array shape nsamples ncomponents Returns the probability each Gaussian state in the model given each sample

sampleselfnsamples1

Generate random samples from the fitted Gaussian distribution

Parameters

nsamples int optional Number of samples to generate Defaults to 1

Returns

Xarray shape nsamples nfeatures Randomly generated sample

yarray shape nsamples Component labels

scoreselfXyNone

Compute the persample average loglikelihood of the given data X

Parameters

Xarraylike shape nsamples ndimensions List of nfeaturesdimensional data points

Each row corresponds to a single data point

Returns

loglikelihood float Log likelihood of the Gaussian mixture given X

scoresamples selfX

Compute the weighted log probabilities for each sample

Parameters

Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points

Each row corresponds to a single data point

Returns

logprob array shape nsamples Log probabilities of each data point in X

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnmixtureBayesianGaussianMixture

•Gaussian Mixture Model Ellipsoids

•Gaussian Mixture Model Sine Curve

•Concentration Prior Type Analysis of Variation Bayesian Gaussian Mixture

2062 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

6252sklearnmixture GaussianMixture

classsklearnmixture GaussianMixture ncomponents1 covariancetype'full' tol0001

regcovar1e06 maxiter100 ninit1

initparams'kmeans' weightsinitNone

meansinitNone precisionsinitNone ran

domstateNone warmstartFalse verbose0

verboseinterval10

Gaussian Mixture

Representation of a Gaussian mixture model probability distribution This class allows to estimate the parameters of a Gaussian mixture distribution

Read more in the User Guide

New in version 018

Parameters

ncomponents int defaults to 1 The number of mixture components

covariancetype 'full' default 'tied' 'diag' 'spherical' String describing the type of

covariance parameters to use Must be one of

'full' each component has its own general covariance matrix

'tied' all components share the same general covariance matrix

'diag' each component has its own diagonal covariance matrix

'spherical' each component has its own single variance

tolfloat defaults to 1e3 The convergence threshold EM iterations will stop when the lower bound average gain is below this threshold

regcovar float defaults to 1e6 Nonnegative regularization added to the diagonal of co

variance Allows to assure that the covariance matrices are all positive

maxiter int defaults to 100 The number of EM iterations to perform

ninit int defaults to 1 The number of initializations to perform The best results are kept

initparams 'kmeans' 'random' defaults to 'kmeans' The method used to initialize the

weights the means and the precisions Must be one of

kmeans responsibilities are initialized using kmeans

random responsibilities are initialized randomly

weightsinit arraylike shape ncomponents optional The userprovided initial weights

defaults to None If it None weights are initialized using the initparams method

meansinit arraylike shape ncomponents nfeatures optional The userprovided initial

means defaults to None If it None means are initialized using the initparams method

precisionsinit arraylike optional The userprovided initial precisions inverse of the co

variance matrices defaults to None If it None precisions are initialized using the

'initparams' method The shape depends on 'covariancetype'

ncomponents ifspherical

nfeatures nfeatures iftied

ncomponents nfeatures ifdiag

ncomponents nfeatures nfeatures iffull

625sklearnmixture Gaussian Mixture Models 2063

scikitlearn user guide Release 0213

randomstate int RandomState instance or None optional defaultNone If int ran
domstate is the seed used by the random number generator If RandomState instance
randomstate is the random number generator If None the random number generator is
the RandomState instance used by nprandom
warmstart bool default to False If 'warmstart' is True the solution of the last fitting is
used as initialization for the next call of fit This can speed up convergence when fit is
called several times on similar problems In that case 'ninit' is ignored and only a single
initialization occurs upon the first call See the Glossary
verbose int default to 0 Enable verbose output If 1 then it prints the current initialization
and each iteration step If greater than 1 then it prints also the log probability and the time
needed for each step
verboseinterval int default to 10 Number of iteration done before the next print
Attributes
weights arraylike shape ncomponents The weights of each mixture components
means arraylike shape ncomponents nfeatures The mean of each mixture component
covariances arraylike The covariance of each mixture component The shape depends on
covariancetype
ncomponents ifspherical
nfeatures nfeatures iftied
ncomponents nfeatures ifdiag
ncomponents nfeatures nfeatures iffull
precisions arraylike The precision matrices for each component in the mixture A preci
sion matrix is the inverse of a covariance matrix A covariance matrix is symmetric posi
tive definite so the mixture of Gaussian can be equivalently parameterized by the precision
matrices Storing the precision matrices instead of the covariance matrices makes it more
efficient to compute the loglikelihood of new samples at test time The shape depends on
covariancetype
ncomponents ifspherical
nfeatures nfeatures iftied
ncomponents nfeatures ifdiag
ncomponents nfeatures nfeatures iffull
precisionscholesky arraylike The cholesky decomposition of the precision matrices of
each mixture component A precision matrix is the inverse of a covariance matrix A covari
ance matrix is symmetric positive definite so the mixture of Gaussian can be equivalently
parameterized by the precision matrices Storing the precision matrices instead of the co
variance matrices makes it more efficient to compute the loglikelihood of new samples at
test time The shape depends on covariancetype
ncomponents ifspherical
nfeatures nfeatures iftied
ncomponents nfeatures ifdiag
ncomponents nfeatures nfeatures iffull
converged bool True when convergence was reached in fit False otherwise
niter int Number of step used by the best fit of EM to reach the convergence
lowerbound float Lower bound value on the loglikelihood of the training data with re
spect to the model of the best fit of EM
2064 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

See also
BayesianGaussianMixture Gaussian mixture model fit with a variational inference
Methods

aicself X Akaike information criterion for the current model on
the input X
bicself X Bayesian information criterion for the current model on
the input X
fitself X y Estimate model parameters with the EM algorithm
fitpredict self X y Estimate model parameters using X and predict the la
bels for X

getparams self deep Get parameters for this estimator
predict self X Predict the labels for the data samples in X using trained
model
predictproba self X Predict posterior probability of each component given
the data
sample self nsamples Generate random samples from the fitted Gaussian dis
tribution
score self X y Compute the persample average loglikelihood of the
given data X
scoresamples self X Compute the weighted log probabilities for each sam
ple

setparams self params Set the parameters of this estimator
init self ncomponents1 covariancetype'full' tol0001 regcovar1e06
maxiter100 ninit1 initparams'kmeans' weightsinitNone meansinitNone
precisionsinitNone randomstateNone warmstartFalse verbose0 ver
boseinterval10

aicselfX
Akaike information criterion for the current model on the input X

Parameters
Xarray of shape nsamples ndimensions
Returns
aicfloat The lower the better
bicselfX

Bayesian information criterion for the current model on the input X

Parameters
Xarray of shape nsamples ndimensions
Returns
bicfloat The lower the better
fitselfXyNone

Estimate model parameters with the EM algorithm
The method fits the model ninit times and sets the parameters with which the model has the
largest likelihood or lower bound Within each trial the method iterates between Estep and Mstep
625sklearnmixture Gaussian Mixture Models 2065

scikitlearn user guide Release 0213

formaxiter times until the change of likelihood or lower bound is less than tol otherwise a ConvergenceWarning is raised If warmstart isTrue then ninit is ignored and a single initialization is performed upon the first call Upon consecutive calls training starts where it left off

Parameters
Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points
Each row corresponds to a single data point

Returns
self
fitpredict selfXyNone
Estimate model parameters using X and predict the labels for X
The method fits the model ninit times and sets the parameters with which the model has the largest likelihood or lower bound Within each trial the method iterates between Estep and Mstep for maxiter times until the change of likelihood or lower bound is less than tol otherwise a ConvergenceWarning is raised After fitting it predicts the most probable label for the input data points
New in version 020

Parameters
Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points
Each row corresponds to a single data point
Returns
labels array shape nsamples Component labels
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values

predictselfX
Predict the labels for the data samples in X using trained model
Parameters
Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points
Each row corresponds to a single data point
Returns

labels array shape nsamples Component labels
predictproba selfX
Predict posterior probability of each component given the data
Parameters
Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points
Each row corresponds to a single data point
Returns

scikitlearn user guide Release 0213

resp array shape nsamples ncomponents Returns the probability each Gaussian state in the model given each sample

sampleselfnsamples1

Generate random samples from the fitted Gaussian distribution

Parameters

nsamples int optional Number of samples to generate Defaults to 1

Returns

Xarray shape nsamples nfeatures Randomly generated sample

yarray shape nsamples Component labels

scoreselfXyNone

Compute the persample average loglikelihood of the given data X

Parameters

Xarraylike shape nsamples ndimensions List of nfeaturesdimensional data points

Each row corresponds to a single data point

Returns

loglikelihood float Log likelihood of the Gaussian mixture given X

scoresamples selfX

Compute the weighted log probabilities for each sample

Parameters

Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points

Each row corresponds to a single data point

Returns

logprob array shape nsamples Log probabilities of each data point in X

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnmixtureGaussianMixture

- Comparing different clustering algorithms on toy datasets
- Density Estimation for a Gaussian mixture
- Gaussian Mixture Model Ellipsoids
- Gaussian Mixture Model Selection
- GMM covariances
- Gaussian Mixture Model Sine Curve

625sklearnmixture Gaussian Mixture Models 2067

scikitlearn user guide Release 0213

626sklearnmodelselection Model Selection

User guide See the Crossvalidation evaluating estimator performance Tuning the hyperparameters of an estimator and Learning curve sections for further details

6261 Splitter Classes

modelselectionGroupKFold nsplits Kfold iterator variant with nonoverlapping groups

modelselectionGroupShuffleSplit ShuffleGroupsOut crossvalidation iterator

modelselectionKFold nsplits shuffle KFolds crossvalidator

modelselectionLeaveOneGroupOut Leave One Group Out crossvalidator

modelselectionLeavePGroupsOut ngroups Leave P Groups Out crossvalidator

modelselectionLeaveOneOut LeaveOneOut crossvalidator

modelselectionLeavePOut p LeavePOut crossvalidator

modelselectionPredefinedSplit testfold Predefined split crossvalidator

modelselectionRepeatedKFold nsplits

Repeated KFold cross validator

modelselectionRepeatedStratifiedKFold Repeated Stratified KFold cross validator

modelselectionShuffleSplit nsplits Random permutation crossvalidator

modelselectionStratifiedKFold nsplits

Stratified KFolds crossvalidator

modelselectionStratifiedShuffleSplit Stratified ShuffleSplit crossvalidator

modelselectionTimeSeriesSplit nsplits

Time Series crossvalidator

sklearnmodelselection GroupKFold

classsklearnmodelselection GroupKFold nsplits'warn'

Kfold iterator variant with nonoverlapping groups

The same group will not appear in two different folds the number of distinct groups has to be at least equal to the number of folds

The folds are approximately balanced in the sense that the number of distinct groups is approximately the same in each fold

Parameters

nsplits int default3 Number of folds Must be at least 2

Changed in version 020 nsplits default value will change from 3 to 5 in v022

See also

LeaveOneGroupOut For splitting the data according to explicit domainspecific stratification of the dataset

Examples

```
import numpy as np
from sklearnmodelselection import GroupKFold
X nparray1 2 3 4 5 6 7 8
y nparray1 2 3 4
groups nparray0 0 2 2
```

2068 Chapter 6 API (Application Programming Interface) Reference

```
scikitlearn user guide Release 0213
groupkfold GroupKFoldnplits2
groupkfoldgetnsplitsX y groups
2
printgroupkfold
GroupKFoldnplits2
for trainindex testindex ingroupkfoldsplitX y groups
printTRAIN trainindex TEST testindex
Xtrain Xtest Xtrainindex Xtestindex
ytrain ytest ytrainindex ytestindex
printXtrain Xtest ytrain ytest

TRAIN 0 1 TEST 2 3
1 2
3 4 5 6
7 8 1 2 3 4
TRAIN 2 3 TEST 0 1
5 6
7 8 1 2
3 4 3 4 1 2
Methods
getnsplits self X y groups Returns the number of splitting iterations in the cross
validator
split self X y groups Generate indices to split data into training and test set
init selfnsplits'warn'
getnsplits selfXNone yNone groupsNone
Returns the number of splitting iterations in the crossvalidator
Parameters
Xobject Always ignored exists for compatibility
yobject Always ignored exists for compatibility
groups object Always ignored exists for compatibility
Returns
nsplits int Returns the number of splitting iterations in the crossvalidator
splitselfXyNone groupsNone
Generate indices to split data into training and test set
Parameters
Xarraylike shape nsamples nfeatures Training data where nsamples is the number
of samples and nfeatures is the number of features
yarraylike shape nsamples optional The target variable for supervised learning prob
lems
groups arraylike with shape nsamples Group labels for the samples used while split
ting the dataset into traintest set
Yields
626sklearnmodelselection Model Selection 2069
```

scikitlearn user guide Release 0213

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

Examples using sklearnmodelselectionGroupKFold

•Visualizing crossvalidation behavior in scikitlearn

sklearnmodelselection GroupShuffleSplit

classsklearnmodelselection GroupShuffleSplit nsplits5 testsizeNone

trainsizeNone randomstateNone

ShuffleGroupsOut crossvalidation iterator

Provides randomized traintest indices to split data according to a thirdparty provided group This group information can be used to encode arbitrary domain specific stratifications of the samples as integers

For instance the groups could be the year of collection of the samples and thus allow for crossvalidation against timebased splits

The difference between LeavePGroupsOut and GroupShuffleSplit is that the former generates splits using all subsets of size punique groups whereas GroupShuffleSplit generates a userdetermined number of random test splits each with a userdetermined fraction of unique groups

For example a less computationally intensive alternative to LeavePGroupsOutp10 would be

GroupShuffleSplittestsize10 nsplits100

Note The parameters testsize andtrainsize refer to groups and not to samples as in ShuffleSplit

Parameters

nsplits int default 5 Number of reshuffling splitting iterations

testsize float int None optional defaultNone If float should be between 00 and 10

and represent the proportion of the dataset to include in the test split If int represents the absolute number of test groups If None the value is set to the complement of the train size

Iftrainsize is also None it will be set to 02

trainsize float int or None default is None If float should be between 00 and 10 and

represent the proportion of the groups to include in the train split If int represents the absolute number of train groups If None the value is automatically set to the complement of the test size

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross

validator

split self X y groups Generate indices to split data into training and test set

init selfnsplits5 testsizeNone trainsizeNone randomstateNone

2070 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getnsplits selfXNone yNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xobject Always ignored exists for compatibility

yobject Always ignored exists for compatibility

groups object Always ignored exists for compatibility

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXyNone groupsNone

Generate indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number

of samples and nfeatures is the number of features

yarraylike shape nsamples optional The target variable for supervised learning prob

lems

groups arraylike with shape nsamples Group labels for the samples used while split

ting the dataset into traintest set

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

Notes

Randomized CV (Cross-Validation) splitters may return different results for each call of split You can make the results

identical by setting randomstate to an integer

Examples using sklearnmodelselectionGroupShuffleSplit

•Visualizing crossvalidation behavior in scikitlearn

sklearnmodelselection KFold

classsklearnmodelselection KFoldnsplits'warn' shuffleFalse randomstateNone

KFolds crossvalidator

Provides traintest indices to split data in traintest sets Split dataset into k consecutive folds without shuffling

by default

Each fold is then used once as a validation while the k - 1 remaining folds form the training set

Read more in the User Guide

Parameters

nsplits int default3 Number of folds Must be at least 2

Changed in version 020 nsplits default value will change from 3 to 5 in v022

626sklearnmodelselection Model Selection 2071

scikitlearn user guide Release 0213

shuffle boolean optional Whether to shuffle the data before splitting into batches

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom Used when shuffle True

See also

StratifiedKFold Takes group information into account to avoid building folds with imbalanced class

distributions for binary or multiclass classification tasks

GroupKFold Kfold iterator variant with nonoverlapping groups

RepeatedKFold Repeats KFold n times

Notes

The firstnsamples nsplits folds have size nsamples nsplits 1 other folds have

sizensamples nsplits wherensamples is the number of samples

Randomized CV (Cross-Validation) splitters may return different results for each call of split You can make the results identical

by settingrandomstate to an integer

Examples

```
import numpy as np
from sklearn.model_selection import KFold
X = np.array([1, 2, 3, 4, 1, 2, 3, 4])
y = np.array([1, 2, 3, 4])
kf = KFold(n_splits=2)
kf.get_n_splits(X)
2
print(kf)
KFold(n_splits=2, random_state=None, shuffle=False)
for train_index, test_index in kf.split(X):
    print('TRAIN: %s TEST: %s' % (train_index, test_index))
    X_train, X_test, y_train, y_test = X[train_index], X[test_index], y[train_index], y[test_index]
    print('TRAIN: %s TEST: %s' % (train_index, test_index))
    X_train, X_test, y_train, y_test = X[train_index], X[test_index], y[train_index], y[test_index]
```

Methods

get_n_splits(self, X, y, groups) Returns the number of splitting iterations in the cross validator

split(self, X, y, groups) Generate indices to split data into training and test set

init(self, n_splits='warn', shuffle=False, random_state=None)

get_n_splits(self, X=None, y=None, groups=None)

Returns the number of splitting iterations in the cross validator

Parameters

2072 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xobject Always ignored exists for compatibility

yobject Always ignored exists for compatibility

groups object Always ignored exists for compatibility

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXyNone groupsNone

Generate indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples The target variable for supervised learning problems

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

Examples using sklearnmodelselectionKFold

- Feature agglomeration vs univariate selection

- Gradient Boosting OutofBag estimates

- Crossvalidation on diabetes Dataset Exercise

- Nested versus nonnested crossvalidation

- Visualizing crossvalidation behavior in scikitlearn

sklearnmodelselection LeaveOneGroupOut

classsklearnmodelselection LeaveOneGroupOut

Leave One Group Out crossvalidator

Provides traintest indices to split data according to a thirdparty provided group This group information can be used to encode arbitrary domain specific stratifications of the samples as integers

For instance the groups could be the year of collection of the samples and thus allow for crossvalidation against timebased splits

Read more in the User Guide

Examples

```
import numpy as np
```

```
from sklearnmodelselection import LeaveOneGroupOut
```

```
X nparray1 2 3 4 5 6 7 8
```

```
y nparray1 2 1 2
```

```
groups nparray1 1 2 2
```

626sklearnmodelselection Model Selection 2073

scikitlearn user guide Release 0213

logo LeaveOneGroupOut

logogetnsplitsX y groups

2

logogetnsplitsgroupsgroups groups is always required

2

printlogo

LeaveOneGroupOut

for trainindex testindex inlogosplitX y groups

printTRAIN trainindex TEST testindex

Xtrain Xtest Xtrainindex Xtestindex

ytrain ytest ytrainindex ytestindex

printXtrain Xtest ytrain ytest

TRAIN 2 3 TEST 0 1

5 6

7 8 1 2

3 4 1 2 1 2

TRAIN 0 1 TEST 2 3

1 2

3 4 5 6

7 8 1 2 1 2

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross

validator

split self X y groups Generate indices to split data into training and test set

init selfargs kwargs

Initialize self See helptypeself for accurate signature

getnsplits selfXNone yNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xobject Always ignored exists for compatibility

yobject Always ignored exists for compatibility

groups arraylike with shape nsamples Group labels for the samples used while split

ting the dataset into traintest set This 'groups' parameter must always be specified to

calculate the number of splits though the other parameters can be omitted

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXyNone groupsNone

Generate indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number

of samples and nfeatures is the number of features

yarraylike of length nsamples optional The target variable for supervised learning

problems

2074 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

groups arraylike with shape nsamples Group labels for the samples used while splitting the dataset into traintest set

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

sklearnmodelselection LeavePGroupsOut

classsklearnmodelselection LeavePGroupsOut ngroups

Leave P Groups Out crossvalidator

Provides traintest indices to split data according to a thirdparty provided group This group information can be used to encode arbitrary domain specific stratifications of the samples as integers

For instance the groups could be the year of collection of the samples and thus allow for crossvalidation against timebased splits

The difference between LeavePGroupsOut and LeaveOneGroupOut is that the former builds the test sets with all the samples assigned to pdifferent values of the groups while the latter uses samples all assigned the same groups

Read more in the User Guide

Parameters

ngroups int Number of groups p to leave out in the test split

See also

GroupKFold Kfold iterator variant with nonoverlapping groups

Examples

```
import numpy as np
from sklearnmodelselection import LeavePGroupsOut
```

```
X nparray1 2 3 4 5 6
```

```
y nparray1 2 1
```

```
groups nparray1 2 3
```

```
lpgo LeavePGroupsOutngroups2
```

```
lpgogetnsplitsX y groups
```

```
3
```

```
lpgogetnsplitsgroupsgroups groups is always required
```

```
3
```

```
printlpgo
```

```
LeavePGroupsOutngroups2
```

```
for trainindex testindex inlpgosplitX y groups
```

```
printTRAIN trainindex TEST testindex
```

```
Xtrain Xtest Xtrainindex Xtestindex
```

```
ytrain ytest ytrainindex ytestindex
```

```
printXtrain Xtest ytrain ytest
```

```
TRAIN 2 TEST 0 1
```

```
5 6 1 2
```

```
3 4 1 1 2
```

```
TRAIN 1 TEST 0 2
```

```
3 4 1 2
```

```
5 6 2 1 1
```

```
626sklearnmodelselection Model Selection 2075
```

scikitlearn user guide Release 0213

TRAIN 0 TEST 1 2

1 2 3 4

5 6 1 2 1

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross validator

split self X y groups Generate indices to split data into training and test set

init selfngroups

getnsplits selfXNone yNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xobject Always ignored exists for compatibility

yobject Always ignored exists for compatibility

groups arraylike with shape nsamples Group labels for the samples used while splitting the dataset into traintest set This 'groups' parameter must always be specified to calculate the number of splits though the other parameters can be omitted

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXyNone groupsNone

Generate indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yarraylike of length nsamples optional The target variable for supervised learning problems

groups arraylike with shape nsamples Group labels for the samples used while splitting the dataset into traintest set

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

sklearnmodelselection LeaveOneOut

classsklearnmodelselection LeaveOneOut

LeaveOneOut crossvalidator

Provides traintest indices to split data in traintest sets Each sample is used once as a test set singleton while the remaining samples form the training set

NoteLeaveOneOut is equivalent to KFoldnsplitsn andLeavePOutp1 wherenis the number of samples

2076 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Due to the high number of test sets which is the same as the number of samples this crossvalidation method can be very costly For large datasets one should favor KFold ShuffleSplit orStratifiedKFold

Read more in the User Guide

See also

LeaveOneGroupOut For splitting the data according to explicit domainspecific stratification of the dataset

GroupKFold Kfold iterator variant with nonoverlapping groups

Examples

```
import numpy as np
from sklearnmodelselection import LeaveOneOut
X nparray1 2 3 4
y nparray1 2
loo LeaveOneOut
loogetnsplitsX
2
printloo
LeaveOneOut
for trainindex testindex inloosplitX
printTRAIN trainindex TEST testindex
Xtrain Xtest Xtrainindex Xtestindex
ytrain ytest ytrainindex ytestindex
printXtrain Xtest ytrain ytest
TRAIN 1 TEST 0
3 4 1 2 2 1
TRAIN 0 TEST 1
1 2 3 4 1 2
```

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross validator

split self X y groups Generate indices to split data into training and test set

init selfargs kwargs

Initialize self See helptypeself for accurate signature

getnsplits selfXyNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yobject Always ignored exists for compatibility

groups object Always ignored exists for compatibility

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

626sklearnmodelselection Model Selection 2077

scikitlearn user guide Release 0213

splitselfXyNone groupsNone

Generate indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yarraylike of length nsamples The target variable for supervised learning problems

groups arraylike with shape nsamples optional Group labels for the samples used

while splitting the dataset into traintest set

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

sklearnmodelselection LeavePOut

classsklearnmodelselection LeavePOut p

LeavePOut crossvalidator

Provides traintest indices to split data in traintest sets This results in testing on all distinct samples of size p while the remaining n - p samples form the training set in each iteration

NoteLeavePOutp is NOT equivalent to KFoldnsplitsnsamples - p which creates non overlapping test sets

Due to the high number of iterations which grows combinatorically with the number of samples this cross validation method can be very costly For large datasets one should favor KFold StratifiedKFold or

ShuffleSplit

Read more in the User Guide

Parameters

pint Size of the test sets Must be strictly greater than the number of samples

Examples

```
import numpy as np
from sklearnmodelselection import LeavePOut
```

```
X nparray1 2 3 4 5 6 7 8
```

```
y nparray1 2 3 4
```

```
lpo LeavePOut2
```

```
lpogetnsplitsX
```

```
6
```

```
printlpo
```

```
LeavePOutp2
```

```
for trainindex testindex inlposplitX
```

```
printTRAIN trainindex TEST testindex
```

```
Xtrain Xtest Xtrainindex Xtestindex
```

```
ytrain ytest ytrainindex ytestindex
```

```
TRAIN 2 3 TEST 0 1
```

```
TRAIN 1 3 TEST 0 2
```

```
TRAIN 1 2 TEST 0 3
```

```
TRAIN 0 3 TEST 1 2
```

2078 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

TRAIN 0 2 TEST 1 3

TRAIN 0 1 TEST 2 3

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross validator

split self X y groups Generate indices to split data into training and test set

init selfp

getnsplits selfXyNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yobject Always ignored exists for compatibility

groups object Always ignored exists for compatibility

splitselfXyNone groupsNone

Generate indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yarraylike of length nsamples The target variable for supervised learning problems

groups arraylike with shape nsamples optional Group labels for the samples used

while splitting the dataset into traintest set

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

sklearnmodelselection PredefinedSplit

classssklearnmodelselection PredefinedSplit testfold

Predefined split crossvalidator

Provides traintest indices to split data into traintest sets using a predefined scheme specified by the user with thetestfold parameter

Read more in the User Guide

Parameters

testfold arraylike shape nsamples The entry testfoldi represents the index of the test set that sample ibelongs to It is possible to exclude sample ifrom any test set ie include sample iin every training set by setting testfoldi equal to 1

626sklearnmodelselection Model Selection 2079

scikitlearn user guide Release 0213

Examples

```
import numpy as np
from sklearnmodelselection import PredefinedSplit
X nparray1 2 3 4 1 2 3 4
y nparray0 0 1 1
testfold 0 1 1 1
ps PredefinedSplittestfold
psgetnsplits
2
```

printps

```
PredefinedSplittestfoldarray 0 1 1 1
for trainindex testindex inpssplit
printTRAIN trainindex TEST testindex
Xtrain Xtest Xtrainindex Xtestindex
ytrain ytest ytrainindex ytestindex
TRAIN 1 2 3 TEST 0
TRAIN 0 2 TEST 1 3
```

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross validator

split self X y groups Generate indices to split data into training and test set

init selftestfold

getnsplits selfXNone yNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xobject Always ignored exists for compatibility

yobject Always ignored exists for compatibility

groups object Always ignored exists for compatibility

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXNone yNone groupsNone

Generate indices to split data into training and test set

Parameters

Xobject Always ignored exists for compatibility

yobject Always ignored exists for compatibility

groups object Always ignored exists for compatibility

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

scikitlearn user guide Release 0213

sklearnmodelselection RepeatedKFold

classsklearnmodelselection RepeatedKFold nsplits5 nrepeats10 randomstateNone

Repeated KFold cross validator

Repeats KFold n times with different randomization in each repetition

Read more in the User Guide

Parameters

nsplits int default5 Number of folds Must be at least 2

nrepeats int default10 Number of times crossvalidator needs to be repeated

randomstate int RandomState instance or None optional defaultNone If int random

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

See also

RepeatedStratifiedKFold Repeats Stratified KFold n times

Notes

Randomized CV (Cross-Validation) splitters may return different results for each call of split You can make the results identical

by settingrandomstate to an integer

Examples

```
import numpy as np
```

```
from sklearnmodelselection import RepeatedKFold
```

```
X nparray1 2 3 4 1 2 3 4
```

```
y nparray0 0 1 1
```

```
rkf RepeatedKFoldnsplits2 nrepeats2 randomstate2652124
```

```
for trainindex testindex inrkfsplitX
```

```
printTRAIN trainindex TEST testindex
```

```
Xtrain Xtest Xtrainindex Xtestindex
```

```
ytrain ytest ytrainindex ytestindex
```

```
TRAIN 0 1 TEST 2 3
```

```
TRAIN 2 3 TEST 0 1
```

```
TRAIN 1 2 TEST 0 3
```

```
TRAIN 0 3 TEST 1 2
```

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross validator

split self X y groups Generates indices to split data into training and test set

init selfnsplits5 nrepeats10 randomstateNone

626sklearnmodelselection Model Selection 2081

scikitlearn user guide Release 0213

getnsplits selfXNone yNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xobject Always ignored exists for compatibility npzerosnsamples may be used as a placeholder

yobject Always ignored exists for compatibility npzerosnsamples may be used as a placeholder

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXyNone groupsNone

Generates indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yarraylike of length nsamples The target variable for supervised learning problems

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

sklearnmodelselection RepeatedStratifiedKFold

classsklearnmodelselection RepeatedStratifiedKFold nsplits5 nrepeats10 randomstateNone

Repeated Stratified KFold cross validator

Repeats Stratified KFold n times with different randomization in each repetition

Read more in the User Guide

Parameters

nsplits int default5 Number of folds Must be at least 2

nrepeats int default10 Number of times crossvalidator needs to be repeated

randomstate None int or RandomState defaultNone Random state to be used to generate random state for each repetition

See also

RepeatedKFold Repeats KFold n times

2082 Chapter 6 API (Application Programming Interface) Reference

Notes

Randomized CV (Cross-Validation) splitters may return different results for each call of split You can make the results identical by settingrandomstate to an integer

Examples

```
import numpy as np
from sklearnmodelselection import RepeatedStratifiedKFold
X nparray1 2 3 4 1 2 3 4
y nparray0 0 1 1
rskf RepeatedStratifiedKFoldnsplits2 nrepeats2
randomstate36851234
for trainindex testindex inrskfsplitX y
printTRAIN trainindex TEST testindex
Xtrain Xtest Xtrainindex Xtestindex
ytrain ytest ytrainindex ytestindex
```

TRAIN 1 2 TEST 0 3
TRAIN 0 3 TEST 1 2
TRAIN 1 3 TEST 0 2
TRAIN 0 2 TEST 1 3

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross validator

split self X y groups Generates indices to split data into training and test set

init selfnsplits5 nrepeats10 randomstateNone

getnsplits selfXNone yNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xobject Always ignored exists for compatibility npzerosnsamples may be used as a placeholder

yobject Always ignored exists for compatibility npzerosnsamples may be used as a placeholder

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXyNone groupsNone

Generates indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

scikitlearn user guide Release 0213

yarraylike of length nsamples The target variable for supervised learning problems
groups arraylike with shape nsamples optional Group labels for the samples used
while splitting the dataset into traintest set

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

sklearnmodelselection ShuffleSplit

class sklearnmodelselection ShuffleSplit nsplits10 testsizeNone trainsizeNone

randomstateNone

Random permutation crossvalidator

Yields indices to split data into training and test sets

Note contrary to other crossvalidation strategies random splits do not guarantee that all folds will be different
although this is still very likely for sizeable datasets

Read more in the User Guide

Parameters

nsplits int default 10 Number of reshuffling splitting iterations

testsize float int None defaultNone If float should be between 00 and 10 and repre
sent the proportion of the dataset to include in the test split If int represents the absolute
number of test samples If None the value is set to the complement of the train size If
trainsize is also None it will be set to 01

trainsize float int or None defaultNone If float should be between 00 and 10 and repre
sent the proportion of the dataset to include in the train split If int represents the absolute
number of train samples If None the value is automatically set to the complement of the
test size

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is
the RandomState instance used by nprandom

Examples

import numpy as np

from sklearnmodelselection import ShuffleSplit

X nparray1 2 3 4 5 6 7 8 3 4 5 6

y nparray1 2 1 2 1 2

rs ShuffleSplitnsplits5 testsize25 randomstate0

rsgetnsplitsX

5

printrs

ShuffleSplitnsplits5 randomstate0 testsize025 trainsizeNone

for trainindex testindex inrssplitX

printTRAIN trainindex TEST testindex

TRAIN 1 3 0 4 TEST 5 2

TRAIN 4 0 2 5 TEST 1 3

2084 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

```
TRAIN 1 2 4 0 TEST 3 5
TRAIN 3 4 1 0 TEST 5 2
TRAIN 3 5 1 0 TEST 2 4
rs ShuffleSplitnplits5 trainsize05 testsize25
randomstate0
for trainindex testindex inrssplitX
printTRAIN trainindex TEST testindex
```

```
TRAIN 1 3 0 TEST 5 2
TRAIN 4 0 2 TEST 1 3
TRAIN 1 2 4 TEST 3 5
TRAIN 3 4 1 TEST 5 2
TRAIN 3 5 1 TEST 2 4
```

Methods
getnsplits self X y groups Returns the number of splitting iterations in the cross validator
split self X y groups Generate indices to split data into training and test set
init selfnsplits10 testsizeNone trainsizeNone randomstateNone
getnsplits selfXNone yNone groupsNone
Returns the number of splitting iterations in the crossvalidator

Parameters
Xobject Always ignored exists for compatibility
yobject Always ignored exists for compatibility
groups object Always ignored exists for compatibility
Returns
nsplits int Returns the number of splitting iterations in the crossvalidator
splitselfXyNone groupsNone

Generate indices to split data into training and test set
Parameters
Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features
yarraylike shape nsamples The target variable for supervised learning problems
groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set
Yields
train ndarray The training set indices for that split
test ndarray The testing set indices for that split
626sklearnmodelselection Model Selection 2085

scikitlearn user guide Release 0213

Notes

Randomized CV (Cross-Validation) splitters may return different results for each call of split You can make the results identical by setting randomstate to an integer

Examples using sklearnmodelselectionShuffleSplit

- Visualizing crossvalidation behavior in scikitlearn

- Plotting Learning Curves

- Scaling the regularization parameter for SVCs

sklearnmodelselection StratifiedKFold

class sklearnmodelselection StratifiedKFold nsplits'warn' shuffleFalse randomstateNone

Stratified KFold crossvalidator

Provides train test indices to split data in train test sets

This crossvalidation object is a variation of KFold that returns stratified folds The folds are made by preserving the percentage of samples for each class

Read more in the User Guide

Parameters

nsplits int default3 Number of folds Must be at least 2

Changed in version 020 nsplits default value will change from 3 to 5 in v022

shuffle boolean optional Whether to shuffle each class's samples before splitting into batches

randomstate int RandomState instance or None optional defaultNone If int random

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom Used when shuffle True

See also

RepeatedStratifiedKFold Repeats Stratified KFold n times

Notes

Train and test sizes may be different in each fold with a difference of at most nclasses

Examples

```
import numpy as np
```

```
from sklearnmodelselection import StratifiedKFold
```

```
X nparray1 2 3 4 1 2 3 4
```

```
y nparray0 0 1 1
```

```
skf StratifiedKFoldnsplits2
```

```
skfgetnsplitsX y
```

```
2
```

2086 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

```
printskf
StratifiedKFoldnplits2 randomstateNone shuffleFalse
  for trainindex testindex inskfsplitX y
printTRAIN trainindex TEST testindex
Xtrain Xtest Xtrainindex Xtestindex
ytrain ytest ytrainindex ytestindex
TRAIN 1 3 TEST 0 2
TRAIN 0 2 TEST 1 3
```

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross validator

split self X y groups Generate indices to split data into training and test set

init selfnplits'warn' shuffleFalse randomstateNone

getnsplits selfXNone yNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xobject Always ignored exists for compatibility

yobject Always ignored exists for compatibility

groups object Always ignored exists for compatibility

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXygroupsNone

Generate indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

Note that providing yis sufficient to generate the splits and hence np

zerosnsamples may be used as a placeholder for Xinstead of actual training data

yarraylike shape nsamples The target variable for supervised learning problems

Stratification is done based on the y labels

groups object Always ignored exists for compatibility

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

Notes

Randomized CV (Cross-Validation) splitters may return different results for each call of split You can make the results identical by setting randomstate to an integer

626sklearnmodelselection Model Selection 2087

scikitlearn user guide Release 0213

Examples using sklearnmodelselectionStratifiedKFold

- Recursive feature elimination with crossvalidation
- Test with permutations the significance of a classification score
- GMM covariances
- Receiver Operating Characteristic ROC with cross validation
- Visualizing crossvalidation behavior in scikitlearn

sklearnmodelselection StratifiedShuffleSplit

classsklearnmodelselection StratifiedShuffleSplit nsplits10 testsizeNone

trainsizeNone ran

domstateNone

Stratified ShuffleSplit crossvalidator

Provides traintest indices to split data in traintest sets

This crossvalidation object is a merge of StratifiedKFold and ShuffleSplit which returns stratified randomized

folds The folds are made by preserving the percentage of samples for each class

Note like the ShuffleSplit strategy stratified random splits do not guarantee that all folds will be different

although this is still very likely for sizeable datasets

Read more in the User Guide

Parameters

nsplits int default 10 Number of reshuffling splitting iterations

testsize float int None optional defaultNone If float should be between 00 and 10

and represent the proportion of the dataset to include in the test split If int represents the

absolute number of test samples If None the value is set to the complement of the train

size Iftrainsize is also None it will be set to 01

trainsize float int or None default is None If float should be between 00 and 10 and

represent the proportion of the dataset to include in the train split If int represents the

absolute number of train samples If None the value is automatically set to the complement

of the test size

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

Examples

import numpy as np

from sklearnmodelselection import StratifiedShuffleSplit

X nparray1 2 3 4 1 2 3 4 1 2 3 4

y nparray0 0 0 1 1 1

sss StratifiedShuffleSplitnsplits5 testsize05 randomstate0

sssgetnsplitsX y

5

printsss

StratifiedShuffleSplitnsplits5 randomstate0

2088 Chapter 6 API (Application Programming Interface) Reference

```
scikitlearn user guide Release 0213
for trainindex testindex inssssplitX y
printTRAIN trainindex TEST testindex
Xtrain Xtest Xtrainindex Xtestindex
ytrain ytest ytrainindex ytestindex
TRAIN 5 2 3 TEST 4 1 0
TRAIN 5 1 4 TEST 0 2 3
TRAIN 5 0 2 TEST 4 3 1
TRAIN 4 1 0 TEST 2 3 5
TRAIN 0 5 1 TEST 3 4 2
```

Methods
getnsplits self X y groups Returns the number of splitting iterations in the cross validator
split self X y groups Generate indices to split data into training and test set
init selfnsplits10 testsizeNone trainsizeNone randomstateNone
getnsplits selfXNone yNone groupsNone
Returns the number of splitting iterations in the crossvalidator

Parameters
Xobject Always ignored exists for compatibility
yobject Always ignored exists for compatibility
groups object Always ignored exists for compatibility
Returns

nsplits int Returns the number of splitting iterations in the crossvalidator
splitselfXygroupsNone
Generate indices to split data into training and test set

Parameters
Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features
Note that providing yis sufficient to generate the splits and hence np
zerosnsamples may be used as a placeholder for Xinstead of actual training data
yarraylike shape nsamples The target variable for supervised learning problems
Stratification is done based on the y labels
groups object Always ignored exists for compatibility

Yields
train ndarray The training set indices for that split
test ndarray The testing set indices for that split

Notes
Randomized CV (Cross-Validation) splitters may return different results for each call of split You can make the results identical by setting randomstate to an integer
626sklearnmodelselection Model Selection 2089

scikitlearn user guide Release 0213

Examples using sklearnmodelselectionStratifiedShuffleSplit

- Visualizing crossvalidation behavior in scikitlearn
- RBF (Radial Basis Function) SVM (Support Vector Machine) parameters

sklearnmodelselection TimeSeriesSplit

classsklearnmodelselection TimeSeriesSplit nsplits'warn' maxtrainsizeNone

Time Series crossvalidator

Provides traintest indices to split time series data samples that are observed at fixed time intervals in traintest sets In each split test indices must be higher than before and thus shuffling in cross validator is inappropriate

This crossvalidation object is a variation of KFold In the kth split it returns first k folds as train set and the k1th fold as test set

Note that unlike standard crossvalidation methods successive training sets are supersets of those that come before them

Read more in the User Guide

Parameters

nsplits int default3 Number of splits Must be at least 2

Changed in version 020 nsplits default value will change from 3 to 5 in v022

maxtrainsize int optional Maximum size for a single training set

Notes

The training set has size insamples nsplits 1 nsamples nsplits

1 in theith split with a test set of size nsamplesnsplits 1

wherensamples is the number of samples

Examples

```
import numpy as np
from sklearnmodelselection import TimeSeriesSplit
```

```
X nparray1 2 3 4 1 2 3 4 1 2 3 4
```

```
y nparray1 2 3 4 5 6
```

```
tscv TimeSeriesSplitnsplits5
```

```
printtscv
```

```
TimeSeriesSplitmaxtrainsizeNone nsplits5
```

```
for trainindex testindex intscvsplitX
```

```
printTRAIN trainindex TEST testindex
```

```
Xtrain Xtest Xtrainindex Xtestindex
```

```
ytrain ytest ytrainindex ytestindex
```

```
TRAIN 0 TEST 1
```

```
TRAIN 0 1 TEST 2
```

```
TRAIN 0 1 2 TEST 3
```

```
TRAIN 0 1 2 3 TEST 4
```

```
TRAIN 0 1 2 3 4 TEST 5
```

2090 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Methods

getnsplits self X y groups Returns the number of splitting iterations in the cross validator

split self X y groups Generate indices to split data into training and test set

init selfnsplits'warn' maxtrainsizeNone

getnsplits selfXNone yNone groupsNone

Returns the number of splitting iterations in the crossvalidator

Parameters

Xobject Always ignored exists for compatibility

yobject Always ignored exists for compatibility

groups object Always ignored exists for compatibility

Returns

nsplits int Returns the number of splitting iterations in the crossvalidator

splitselfXyNone groupsNone

Generate indices to split data into training and test set

Parameters

Xarraylike shape nsamples nfeatures Training data where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Always ignored exists for compatibility

groups arraylike with shape nsamples Always ignored exists for compatibility

Yields

train ndarray The training set indices for that split

test ndarray The testing set indices for that split

Examples using sklearnmodelselectionTimeSeriesSplit

•Visualizing crossvalidation behavior in scikitlearn

6262 Splitter Functions

modelselectioncheckcv cv y classifier Input checker utility for building a crossvalidator

modelselectiontraintestsplit arrays

Split arrays or matrices into random train and test subsets

sklearnmodelselection checkcv

sklearnmodelselection checkcv cv'warn' yNone classifierFalse

Input checker utility for building a crossvalidator

Parameters

626sklearnmodelselection Model Selection 2091

scikitlearn user guide Release 0213

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation
 - integer to specify the number of folds
 - CV (Cross-Validation) splitter
 - An iterable yielding train test splits as arrays of indices
- For integerNone inputs if classifier is True and yis either binary or multiclass StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value will change from 3fold to 5fold in v022

yarraylike optional The target variable for supervised learning problems

classifier boolean optional default False Whether the task is a classification task in which case stratified KFold will be used

Returns

checkedcv a crossvalidator instance The return value is a crossvalidator which generates the traintest splits via the split method

sklearnmodelselection traintestsplit

sklearnmodelselection traintestsplit arrays options

Split arrays or matrices into random train and test subsets

Quick utility that wraps input validation and nextShuffleSplitsplitX y and application to

input data into a single call for splitting and optionally subsampling data in a oneliner

Read more in the User Guide

Parameters

arrays sequence of indexables with same length shape0 Allowed inputs are lists numpy

arrays scipysparse matrices or pandas dataframes

testsize float int or None optional defaultNone If float should be between 00 and 10

and represent the proportion of the dataset to include in the test split If int represents the absolute number of test samples If None the value is set to the complement of the train

size Iftrainsize is also None it will be set to 025

trainsize float int or None defaultNone If float should be between 00 and 10 and

represent the proportion of the dataset to include in the train split If int represents the absolute number of train samples If None the value is automatically set to the complement of the test size

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

shuffle boolean optional defaultTrue Whether or not to shuffle the data before splitting If

shuffleFalse then stratify must be None

stratify arraylike or None defaultNone If not None data is split in a stratified fashion

using this as the class labels

2092 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

splitting list length2 lenarrays List containing traintest split of inputs

New in version 016 If the input is sparse the output will be a scipysparse

csrmatrix Else output type is the same as the input type

Examples

```
import numpy as np
from sklearnmodelselection import traintestsplit
```

```
X y nparange10reshape5 2 range5
```

```
X
```

```
array0 1
```

```
2 3
```

```
4 5
```

```
6 7
```

```
8 9
```

```
listy
```

```
0 1 2 3 4
```

```
Xtrain Xtest ytrain ytest traintestsplit
```

```
X y testsize033 randomstate42
```

```
Xtrain
```

```
array4 5
```

```
0 1
```

```
6 7
```

```
ytrain
```

```
2 0 3
```

```
Xtest
```

```
array2 3
```

```
8 9
```

```
ytest
```

```
1 4
```

```
traintestsplit shuffle False
```

```
0 1 2 3 4
```

Examples using sklearnmodelselectiontraintestsplit

- Faces recognition example using eigenfaces and SVMs

- Prediction Latency

- Probability Calibration curves

- Probability calibration of classifiers

- Classifier comparison

- Column Transformer with Mixed Types

- Effect of transforming the targets in regression model

- Comparing random forests and the multioutput meta estimator

- Early stopping of Gradient Boosting

626sklearnmodelselection Model Selection 2093

scikitlearn user guide Release 0213

- Feature transformations with ensembles of trees
- Gradient Boosting OutofBag estimates
- Pipeline Anova SVM (Support Vector Machine)
- Comparing various online solvers
- MNIST classffication using multinomial logistic L1
- Multiclass sparse logisitic regression on newgroups20
- Early stopping of Stochastic Gradient Descent
- Parameter estimation using grid search with crossvalidation
- Confusion matrix
- Receiver Operating Characteristic ROC
- PrecisionRecall
- Classifier Chain
- Comparing Nearest Neighbors with and without Neighborhood Components Analysis
- Dimensionality Reduction with Neighborhood Components Analysis
- Restricted Boltzmann Machine features for digit classification
- Varying regularization in Multilayer Perceptron
- Using FunctionTransformer to select columns
- Importance of Feature Scaling
- Map data to a normal distribution
- Feature discretization
- Understanding the decision tree structure

6263 Hyperparameter optimizers
modelselectionGridSearchCV estimator Exhaustive search over specified parameter values for an estimator

modelselectionParameterGrid paramgrid Grid of parameters with a discrete number of values for each

modelselectionParameterSampler Generator on parameters sampled from given distributions

modelselectionRandomizedSearchCV

Randomized search on hyper parameters

sklearnmodelselection GridSearchCV

classssklearnmodelselection GridSearchCV estimator paramgrid scoringNone

njobsNone iid'warn' refitTrue

cv'warn' verbose0 predispatch'2njobs'

errorscore'raisedeprecating' re

turntrainscoreFalse

Exhaustive search over specified parameter values for an estimator

Important members are fit predict

2094 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

GridSearchCV implements a “fit” and a “score” method. It also implements “predict”, “predict_proba”, “decision_function”, “transform” and “inverse_transform” if they are implemented in the estimator used. The parameters of the estimator used to apply these methods are optimized by cross-validated gridsearch over a parameter grid.

Read more in the User Guide

Parameters

estimator: estimator object. This is assumed to implement the scikitlearn estimator interface.

Either estimator needs to provide a score function or scoring must be passed.

param_grid: dict or list of dictionaries. Dictionary with parameters names (string) as keys and lists of parameter settings to try as values or a list of such dictionaries in which case the grids spanned by each dictionary in the list are explored. This enables searching over any sequence of parameter settings.

scoring: string, callable, list/tuple, dict or None. default: None. A single string see The scoring parameter defining model evaluation rules or a callable see Defining your scoring strategy from metric functions to evaluate the predictions on the test set.

For evaluating multiple metrics, either give a list of unique strings or a dict with names as keys and callables as values.

NOTE that when using custom scorers, each scorer should return a single value. Metric functions returning a list/array of values can be wrapped into multiple scorers that return one value each.

See Specifying multiple metrics for evaluation for an example.

If None, the estimator’s score method is used.

n_jobs: int or None. optional. default: None. Number of jobs to run in parallel. None means 1 unless in a joblibparallelbackend context. 1 means using all processors. See Glossary for more details.

pre_dispatch: int or string. optional. Controls the number of jobs that get dispatched during parallel execution. Reducing this number can be useful to avoid an explosion of memory consumption when more jobs get dispatched than CPUs can process. This parameter can be

- None, in which case all the jobs are immediately created and spawned. Use this for lightweight and fast-running jobs to avoid delays due to on-demand spawning of the jobs.
- An int giving the exact number of total jobs that are spawned.
- A string giving an expression as a function of n_jobs as in ‘2*n_jobs’.

iid: boolean. default: ‘warn’. If True, return the average score across folds weighted by the number of samples in each test set. In this case, the data is assumed to be identically distributed across the folds and the loss minimized is the total loss per sample, and not the mean loss across the folds. If False, return the average score across folds. Default is True, but will change to False in version 0.22 to correspond to the standard definition of cross-validation.

Changed in version 0.20: Parameter iid will change from True to False by default in version 0.22 and will be removed in 0.24.

cv: int, cross-validation generator or an iterable. optional. Determines the cross-validation splitting strategy. Possible inputs for cv are:

- None, to use the default 3-fold cross-validation.

626sklearnmodelselection Model Selection 2095

scikitlearn user guide Release 0213

- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs if the estimator is a classifier and yis either binary or multiclass StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here
Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

refit boolean string or callable defaultTrue Refit an estimator using the best found parameters on the whole dataset

For multiple metric evaluation this needs to be a string denoting the scorer that would be used to find the best parameters for refitting the estimator at the end

Where there are considerations other than maximum score in choosing a best estimator torrefit can be set to a function which returns the selected bestindex given

cvresults

The refitted estimator is made available at the bestestimator attribute and permits usingpredict directly on this GridSearchCV instance

Also for multiple metric evaluation the attributes bestindex bestscore and bestparams will only be available if refit is set and all of them will be determined wrt this specific scorer bestscore is not returned if refit is callable

Seescoring parameter to know more about multiple metric evaluation

Changed in version 020 Support for callable added

verbose integer Controls the verbosity the higher the more messages

errorscore 'raise' or numeric Value to assign to the score if an error occurs in estimator fitting If set to 'raise' the error is raised If a numeric value is given FitFailedWarning

is raised This parameter does not affect the refit step which will always raise the error

Default is 'raise' but from version 022 it will change to npnan

returntrainscore boolean defaultFalse If False thecvresults attribute will not

include training scores Computing training scores is used to get insights on how different parameter settings impact the overfittingunderfitting tradeoff However computing the scores on the training set can be computationally expensive and is not strictly required to select the parameters that yield the best generalization performance

Attributes

cvresults dict of numpy masked ndarrays A dict with keys as column headers and values as columns that can be imported into a pandas DataFrame

For instance the below given table

| paramkernel | paramgamma | paramdegree | split0testscore | rankt |
|-------------|------------|-------------|-----------------|-------|
| 'poly' | - 2 | 080 | 2 | |
| 'poly' | - 3 | 070 | 4 | |
| 'rbf' | 01 | - 080 | 3 | |
| 'rbf' | 02 | - 093 | 1 | |

will be represented by a cvresults dict of

2096 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

paramkernel maskedarraydata poly poly rbf rbf
mask False False False False
paramgamma maskedarraydata 01 02
mask True True False False
paramdegree maskedarraydata 20 30
mask False False True True
split0testscore 080 070 080 093
split1testscore 082 050 070 078
meantestscore 081 060 075 085
stdtestscore 001 010 005 008
ranktestscore 2 4 3 1
split0trainscore 080 092 070 093
split1trainscore 082 055 070 087
meantrainscore 081 074 070 090
stdtrainscore 001 019 000 003
meanfittime 073 063 043 049
stdfittime 001 002 001 001
meanscoretime 001 006 004 004
stdscoretime 000 000 000 001
params kernel poly degree 2

NOTE
The keyparams is used to store a list of parameter settings dicts for all the parameter candidates

Themeanfittime stdfittime meanscoretime and stdscoretime are all in seconds
For multimetric evaluation the scores for all the scorers are available in the cvresults dict at the keys ending with that scorer's name scorername instead ofscore shown above 'split0testprecision' 'meantrainprecision' etc
bestestimator estimator or dict Estimator that was chosen by the search ie estimator which gave highest score or smallest loss if specified on the left out data Not available if refitFalse

Seerefit parameter for more information on allowed values
bestscore float Mean crossvalidated score of the bestestimator
For multimetric evaluation this is present only if refit is specified
bestparams dict Parameter setting that gave the best results on the hold out data
For multimetric evaluation this is present only if refit is specified
bestindex int The index of the cvresults arrays which corresponds to the best candidate parameter setting
The dict at searchcvresultsparmssearchbestindex gives the parameter setting for the best model that gives the highest mean score search bestscore
For multimetric evaluation this is present only if refit is specified
scorer function or a dict Scorer function used on the held out data to choose the best parameters for the model
626sklearnmodelselection Model Selection 2097

scikitlearn user guide Release 0213

For multimetric evaluation this attribute holds the validated scoring dict which maps

the scorer key to the scorer callable

nsplits int The number of crossvalidation splits foldsiterations

refittime float Seconds used for refitting the best model on the whole dataset

This is present only if refit is not False

See also

ParameterGrid generates all the combinations of a hyperparameter grid

sklearnmodelselectiontrainstestsplit utility function to split the data into a development

set usable for fitting a GridSearchCV instance and an evaluation set for its final evaluation

sklearnmetricsmakescorer Make a scorer from a performance metric or loss function

Notes

The parameters selected are those that maximize the score of the left out data unless an explicit score is passed

in which case it is used instead

Ifnjobs was set to a value higher than one the data is copied for each point in the grid and not njobs times

This is done for efficiency reasons if individual jobs take very little time but may raise errors if the dataset

is large and not enough memory is available A workaround in this case is to set predispatch Then

the memory is copied only predispatch many times A reasonable value for predispatch is2

njobs

Examples

```
from sklearn import svm datasets
```

```
from sklearnmodelselection import GridSearchCV
```

```
iris datasetsloadiris
```

```
parameters kernellinear rbf C1 10
```

```
svc svmSVCgammascale
```

```
clf GridSearchCVsvc parameters cv5
```

```
clffitirisdata iristarget
```

GridSearchCVcv5 errorscore

estimatorSVCC10 cachesize classweight coef0

decisionfunctionsshapeovr degree gamma

kernelrbf maxiter1 probabilityFalse

randomstateNone shrinkingTrue tol

verboseFalse

iid njobsNone

paramgrid predispatch refit returntrainscore

scoring verbose

sortedclfcvresultskeys

meanfittime meanscoretime meantestscore

paramC paramkernel params

ranktestscore split0testscore

split2testscore

stdfittime stdscoretime stdtestscore

2098 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Methods

decisionfunction self X Call decisionfunction on the estimator with the best found parameters

fitself X y groups Run fit with all sets of parameters

getparams self deep Get parameters for this estimator

inversetransform self Xt Call inversetransform on the estimator with the best found params

predict self X Call predict on the estimator with the best found parameters

predictlogproba self X Call predictlogproba on the estimator with the best found parameters

predictproba self X Call predictproba on the estimator with the best found parameters

score self X y Returns the score on the given data if the estimator has been refit

setparams self params Set the parameters of this estimator

transform self X Call transform on the estimator with the best found parameters

init selfestimator paramgrid scoringNone njobsNone iid'warn' refitTrue cv'warn' verbose0 predispatch'2njobs' errorscore'raisedeprecating' re

turntrainscoreFalse

decisionfunction selfX

Call decisionfunction on the estimator with the best found parameters

Only available if refitTrue and the underlying estimator supports decisionfunction

Parameters

Xindexable length nsamples Must fulfill the input assumptions of the underlying estimator

fitselfXyNone groupsNone fitparams

Run fit with all sets of parameters

Parameters

Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples or nsamples noutput optional Target relative to X for classification or regression None for unsupervised learning

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set Only used in conjunction with a "Group" cv

instance eg GroupKFold

fitparams dict of string object Parameters passed to the fit method of the estimator

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

626sklearnmodelselection Model Selection 2099

scikitlearn user guide Release 0213

Returns
params mapping of string to any Parameter names mapped to their values
inversetransform selfXt
Call inversetransform on the estimator with the best found params
Only available if the underlying estimator implements inversetransform andrefitTrue
Parameters
Xtindexable length nsamples Must fulfill the input assumptions of the underlying estimator
predictselfX
Call predict on the estimator with the best found parameters
Only available if refitTrue and the underlying estimator supports predict
Parameters
Xindexable length nsamples Must fulfill the input assumptions of the underlying estimator
predictlogproba selfX
Call predictlogproba on the estimator with the best found parameters
Only available if refitTrue and the underlying estimator supports predictlogproba
Parameters
Xindexable length nsamples Must fulfill the input assumptions of the underlying estimator
predictproba selfX
Call predictproba on the estimator with the best found parameters
Only available if refitTrue and the underlying estimator supports predictproba
Parameters
Xindexable length nsamples Must fulfill the input assumptions of the underlying estimator
scoreselfXyNone
Returns the score on the given data if the estimator has been refit
This uses the score defined by scoring where provided and the bestestimatorscore method otherwise
Parameters
Xarraylike shape nsamples nfeatures Input data where nsamples is the number of samples and nfeatures is the number of features
yarraylike shape nsamples or nsamples noutput optional Target relative to X for classification or regression None for unsupervised learning
Returns
score float
setparams selfparams
Set the parameters of this estimator
2100 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Call transform on the estimator with the best found parameters

Only available if the underlying estimator supports transform andrefitTrue

Parameters

Xindexable length nsamples Must fulfill the input assumptions of the underlying estimator

Examples using sklearnmodelselectionGridSearchCV

- Comparison of kernel ridge regression and SVR (Support Vector Regression)
- Faces recognition example using eigenfaces and SVMs
- Feature agglomeration vs univariate selection
- Concatenating multiple feature extraction methods
- Pipelining chaining a PCA (Principal Component Analysis) and a logistic regression
- Column Transformer with Mixed Types
- Selecting dimensionality reduction with Pipeline and GridSearchCV
- Shrinkage covariance estimation LedoitWolf vs OAS and maxlikelihood
- Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA
- Crossvalidation on diabetes Dataset Exercise
- Comparison of kernel ridge and Gaussian process regression
- Parameter estimation using grid search with crossvalidation
- Comparing randomized search and grid search for hyperparameter estimation
- Nested versus nonnested crossvalidation
- Demonstration of multimetric evaluation on crossvalscore and GridSearchCV
- Balance model complexity and crossvalidated score
- Sample pipeline for text feature extraction and evaluation
- Kernel Density Estimation
- Feature discretization
- Scaling the regularization parameter for SVCs
- RBF (Radial Basis Function) SVM (Support Vector Machine) parameters

scikitlearn user guide Release 0213

sklearnmodelselection ParameterGrid

classsklearnmodelselection ParameterGrid paramgrid

Grid of parameters with a discrete number of values for each

Can be used to iterate over parameter value combinations with the Python builtin function iter

Read more in the User Guide

Parameters

paramgrid dict of string to sequence or sequence of such The parameter grid to explore as a dictionary mapping estimator parameters to sequences of allowed values

An empty dict signifies default parameters

A sequence of dicts signifies a sequence of grids to search and is useful to avoid exploring parameter combinations that make no sense or have no effect See the examples below

See also

GridSearchCV UsesParameterGrid to perform a full parallelized parameter search

Examples

```
from sklearnmodelselection import ParameterGrid
```

```
paramgrid = {'a': [1, 2], 'b': [True, False]}
```

```
list(ParameterGrid(paramgrid))
```

```
[{'a': 1, 'b': True}, {'a': 1, 'b': False},
```

```
 {'a': 2, 'b': True}, {'a': 2, 'b': False}]
```

```
True
```

```
grid = {'kernel': ['linear', 'rbf'], 'gamma': [1, 10]}
```

```
list(ParameterGrid(grid))
```

```
[{'kernel': 'rbf', 'gamma': 1},
```

```
 {'kernel': 'rbf', 'gamma': 10},
```

```
 {'kernel': 'linear', 'gamma': 1},
```

```
 {'kernel': 'linear', 'gamma': 10}]
```

```
True
```

```
init selfparamgrid
```

sklearnmodelselection ParameterSampler

classsklearnmodelselection ParameterSampler paramdistributions niter randomstate

domstateNone

Generator on parameters sampled from given distributions

Nondeterministic iterable over random candidate combinations for hyper parameter search If all parameters are presented as a list sampling without replacement is performed If at least one parameter is given as a distribution sampling with replacement is used It is highly recommended to use continuous distributions for continuous parameters

Note that before SciPy 0.16 the scipystatsdistributions do not accept a custom RNG instance

and always use the singleton RNG from numpy.random Hence setting randomstate will not guarantee

a deterministic iteration whenever scipystats distributions are used to define the parameter search space

Deterministic behavior is however guaranteed from SciPy 0.16 onwards

2102 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

paramdistributions dict Dictionary where the keys are parameters and values are distributions from which a parameter is to be sampled Distributions either have to provide a rvs function to sample from them or can be given as a list of values where a uniform distribution is assumed

niter integer Number of parameter settings that are produced

randomstate int RandomState instance or None optional defaultNone Pseudo random

number generator state used for random uniform sampling from lists of possible values

instead of scipystats distributions If int randomstate is the seed used by the random

number generator If RandomState instance randomstate is the random number generator

If None the random number generator is the RandomState instance used by nprandom

Returns

params dict of string to any Yields dictionaries mapping each estimator parameter to as sam

pled value

Examples

```
from sklearnmodelselection import ParameterSampler
```

```
from scipystatsdistributions import expon
```

```
import numpy as np
```

```
rng = nprandom.RandomState(0)
```

```
paramgrid = a1 2 b expon
```

```
paramlist = list(ParameterSampler(paramgrid, niter=4,
```

```
    randomstate=rng
```

```
roundedlist = dict(roundv=6, fork=vinditems,
```

```
    for dinparamlist
```

```
roundedlist = b 089856 a 1
```

```
    b 0923223 a 1
```

```
    b 1878964 a 2
```

```
    b 1038159 a 2
```

```
True
```

```
init selfparamdistributions niter randomstate=None
```

```
sklearnmodelselection RandomizedSearchCV
```

```
class sklearnmodelselection RandomizedSearchCV estimator paramdistributions
```

```
niter=10 scoring=None
```

```
njobs=None iid='warn' re
```

```
fit=True cv='warn' ver
```

```
bose=0 predispatch='2njobs'
```

```
randomstate=None
```

```
errorscore='raisedeprecating' re
```

```
turntrainscore=False
```

Randomized search on hyper parameters

RandomizedSearchCV implements a “fit” and a “score” method It also implements “predict” “predictproba”

“decisionfunction” “transform” and “inversetransform” if they are implemented in the estimator used

626sklearnmodelselection Model Selection 2103

scikitlearn user guide Release 0213

The parameters of the estimator used to apply these methods are optimized by crossvalidated search over parameter settings

In contrast to GridSearchCV not all parameter values are tried out but rather a fixed number of parameter settings is sampled from the specified distributions The number of parameter settings that are tried is given by niter

If all parameters are presented as a list sampling without replacement is performed If at least one parameter is given as a distribution sampling with replacement is used It is highly recommended to use continuous distributions for continuous parameters

Note that before SciPy 016 the scipystatsdistributions do not accept a custom RNG instance and always use the singleton RNG from numpyrandom Hence setting randomstate will not guarantee a deterministic iteration whenever scipystats distributions are used to define the parameter search space Read more in the User Guide

Parameters

estimator estimator object A object of that type is instantiated for each grid point This is assumed to implement the scikitlearn estimator interface Either estimator needs to provide ascore function or scoring must be passed

paramdistributions dict Dictionary with parameters names string as keys and distributions or lists of parameters to try Distributions must provide a rvs method for sampling such as those from scipystatsdistributions If a list is given it is sampled uniformly

niter int default10 Number of parameter settings that are sampled niter trades off runtime vs quality of the solution

scoring string callable listtuple dict or None default None A single string see The scoring parameter defining model evaluation rules or a callable see Defining your scoring strategy from metric functions to evaluate the predictions on the test set

For evaluating multiple metrics either give a list of unique strings or a dict with names as keys and callables as values

NOTE that when using custom scorers each scorer should return a single value Metric functions returning a listarray of values can be wrapped into multiple scorers that return one value each

SeeSpecifying multiple metrics for evaluation for an example

If None the estimator’s score method is used

njobs int or None optional defaultNone Number of jobs to run in parallel None means 1 unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

predispach int or string optional Controls the number of jobs that get dispatched during parallel execution Reducing this number can be useful to avoid an explosion of memory consumption when more jobs get dispatched than CPUs can process This parameter can be

- None in which case all the jobs are immediately created and spawned Use this for lightweight and fastrunning jobs to avoid delays due to ondemand spawning of the jobs
- An int giving the exact number of total jobs that are spawned
- A string giving an expression as a function of njobs as in ‘2njobs’

scikitlearn user guide Release 0213

iidboolean default'warn' If True return the average score across folds weighted by the number of samples in each test set In this case the data is assumed to be identically distributed across the folds and the loss minimized is the total loss per sample and not the mean loss across the folds If False return the average score across folds Default is True but will change to False in version 022 to correspond to the standard definition of cross validation

Changed in version 020 Parameter iid will change from True to False by default in version 022 and will be removed in 024

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold cross validation
- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs if the estimator is a classifier and yis either binary or multiclass StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

refit boolean string or callable defaultTrue Refit an estimator using the best found parameters on the whole dataset

For multiple metric evaluation this needs to be a string denoting the scorer that would be used to find the best parameters for refitting the estimator at the end

Where there are considerations other than maximum score in choosing a best estimator refit can be set to a function which returns the selected bestindex given the cvresults

The refitted estimator is made available at the bestestimator attribute and permits usingpredict directly on this RandomizedSearchCV instance

Also for multiple metric evaluation the attributes bestindex bestscore and bestparams will only be available if refit is set and all of them will be determined wrt this specific scorer When refit is callable bestscore is disabled

Seescoring parameter to know more about multiple metric evaluation

Changed in version 020 Support for callable added

verbose integer Controls the verbosity the higher the more messages

randomstate int RandomState instance or None optional defaultNone Pseudo random number generator state used for random uniform sampling from lists of possible values instead of scipystats distributions If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom errorscore 'raise' or numeric Value to assign to the score if an error occurs in estimator fitting If set to 'raise' the error is raised If a numeric value is given FitFailedWarning is raised This parameter does not affect the refit step which will always raise the error Default is 'raise' but from version 022 it will change to npnan

626sklearnmodelselection Model Selection 2105

scikitlearn user guide Release 0213

returntrainscore boolean defaultFalse If False thecvresults attribute will not include training scores Computing training scores is used to get insights on how different parameter settings impact the overfittingunderfitting tradeoff However computing the scores on the training set can be computationally expensive and is not strictly required to select the parameters that yield the best generalization performance

Attributes
cvresults dict of numpy masked ndarrays A dict with keys as column headers and values as columns that can be imported into a pandas DataFrame

For instance the below given table
paramkernel paramgamma split0testscore ranktestscore
'rbf' 01 080 2
'rbf' 02 090 1
'rbf' 03 070 1
will be represented by a cvresults dict of

```
paramkernel maskedarraydata rbf rbf rbf
mask False
paramgamma maskedarraydata 01 02 03 mask False
split0testscore 080 090 070
split1testscore 082 050 070
meantestscore 081 070 070
stdtestscore 001 020 000
ranktestscore 3 1 1
split0trainscore 080 092 070
split1trainscore 082 055 070
meantrainscore 081 074 070
stdtrainscore 001 019 000
meanfittime 073 063 043
stdfittime 001 002 001
meanscoretime 001 006 004
stdscoretime 000 000 000
params kernel rbf gamma 01
```

NOTE
The keyparams is used to store a list of parameter settings dicts for all the parameter candidates

Themeanfittime stdfittime meanscoretime and stdscoretime are all in seconds
For multimetric evaluation the scores for all the scorers are available in the cvresults dict at the keys ending with that scorer's name scorername instead ofscore shown above 'split0testprecision' 'meantrainprecision' etc
bestestimator estimator or dict Estimator that was chosen by the search ie estimator which gave highest score or smallest loss if specified on the left out data Not available if refitFalse

For multimetric evaluation this attribute is present only if refit is specified
Seerefit parameter for more information on allowed values
2106 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

bestscore float Mean crossvalidated score of the bestestimator

For multimetric evaluation this is not available if refit isFalse Seerefit parameter for more information

bestparams dict Parameter setting that gave the best results on the hold out data

For multimetric evaluation this is not available if refit isFalse Seerefit parameter for more information

bestindex int The index of the cvresults arrays which corresponds to the best candidate parameter setting

The dict at searchcvresultsparmssearchbestindex gives

the parameter setting for the best model that gives the highest mean score search bestscore

For multimetric evaluation this is not available if refit isFalse Seerefit parameter for more information

scorer function or a dict Scorer function used on the held out data to choose the best parameters for the model

For multimetric evaluation this attribute holds the validated scoring dict which maps the scorer key to the scorer callable

nsplits int The number of crossvalidation splits foldsiterations

refittime float Seconds used for refitting the best model on the whole dataset This is present only if refit is not False

See also

GridSearchCV Does exhaustive search over a grid of parameters

ParameterSampler A generator over parameter settings constructed from paramdistributions Notes

The parameters selected are those that maximize the score of the heldout data according to the scoring parameter

Ifnjobs was set to a value higher than one the data is copied for each parameter settingand not njobs times This is done for efficiency reasons if individual jobs take very little time but may raise errors if the dataset

is large and not enough memory is available A workaround in this case is to set predispatch Then the memory is copied only predispatch many times A reasonable value for predispatch is2

njobs

Methods

decisionfunction self X Call decisionfunction on the estimator with the best found parameters

fitself X y groups Run fit with all sets of parameters

getparams self deep Get parameters for this estimator

inversetransform self Xt Call inversetransform on the estimator with the best found params

Continued on next page

626sklearnmodelselection Model Selection 2107

predict self X Call predict on the estimator with the best found parameters

predictlogproba self X Call predictlogproba on the estimator with the best found parameters

predictproba self X Call predictproba on the estimator with the best found parameters

score self X y Returns the score on the given data if the estimator has been refit

setparams self params Set the parameters of this estimator

transform self X Call transform on the estimator with the best found parameters

init selfestimator paramdistributions niter10 scoringNone njobsNone iid'warn'

refitTrue cv'warn' verbose0 predispatch'2njobs' randomstateNone

errorscore'raisedeprecating' returntrainscoreFalse

decisionfunction selfX

Call decisionfunction on the estimator with the best found parameters

Only available if refitTrue and the underlying estimator supports decisionfunction

Parameters

Xindexable length nsamples Must fulfill the input assumptions of the underlying estimator

fitselfXyNone groupsNone fitparams

Run fit with all sets of parameters

Parameters

Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples or nsamples noutput optional Target relative to X for classification or regression None for unsupervised learning

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set Only used in conjunction with a "Group" cv instance eg GroupKFold

fitparams dict of string object Parameters passed to the fit method of the estimator

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfXt

Call inversetransform on the estimator with the best found params

Only available if the underlying estimator implements inversetransform andrefitTrue

Parameters

2108 Chapter 6 API (Application Programming Interface) Reference

Xtindexable length nsamples Must fulfill the input assumptions of the underlying estimator

predictselfX

Call predict on the estimator with the best found parameters

Only available if refitTrue and the underlying estimator supports predict

Parameters

Xtindexable length nsamples Must fulfill the input assumptions of the underlying estimator

predictlogproba selfX

Call predictlogproba on the estimator with the best found parameters

Only available if refitTrue and the underlying estimator supports predictlogproba

Parameters

Xtindexable length nsamples Must fulfill the input assumptions of the underlying estimator

predictproba selfX

Call predictproba on the estimator with the best found parameters

Only available if refitTrue and the underlying estimator supports predictproba

Parameters

Xtindexable length nsamples Must fulfill the input assumptions of the underlying estimator

scoreselfXyNone

Returns the score on the given data if the estimator has been refit

This uses the score defined by scoring where provided and the bestestimatorscore method

otherwise

Parameters

Xarraylike shape nsamples nfeatures Input data where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples or nsamples noutput optional Target relative to X

for classification or regression None for unsupervised learning

Returns

score float

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Call transform on the estimator with the best found parameters

Only available if the underlying estimator supports transform andrefitTrue

scikitlearn user guide Release 0213

Parameters

Xindexable length nsamples Must fulfill the input assumptions of the underlying estimator

Examples using sklearnmodelselectionRandomizedSearchCV

•Comparing randomized search and grid search for hyperparameter estimation

modelselectionfitgridpoint X y

Run fit on one set of parameters

sklearnmodelselection fitgridpoint

sklearnmodelselection fitgridpoint Xyestimator parameters train testscorer

verbose errorscore'raisedeprecating'

fitparams

Run fit on one set of parameters

Parameters

Xarraylike sparse matrix or list Input data

yarraylike or None Targets for input data

estimator estimator object A object of that type is instantiated for each grid point This is assumed to implement the scikitlearn estimator interface Either estimator needs to provide a score function or scoring must be passed

parameters dict Parameters to be set on estimator for this grid point

train ndarray dtype int or bool Boolean mask or indices for training set

test ndarray dtype int or bool Boolean mask or indices for test set

scorer callable or None The scorer callable object function must have its signature as

scorerestimator X y

IfNone the estimator's score method is used

verbose int Verbosity level

fitparams kwargs Additional parameter passed to the fit function of the estimator

errorscore 'raise' or numeric Value to assign to the score if an error occurs in estimator

fitting If set to 'raise' the error is raised If a numeric value is given FitFailedWarning

is raised This parameter does not affect the refit step which will always raise the error

Default is 'raise' but from version 022 it will change to npnan

Returns

score float Score of this parameter setting on given test split

parameters dict The parameters that have been evaluated

nsamplestest int Number of test samples in this split

6264 Model validation

2110 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

modelselectioncrossvalidate estimator

XEvaluate metrics by crossvalidation and also record
fitscore times

modelselectioncrossvalpredict estimator

XGenerate crossvalidated estimates for each input data point

modelselectioncrossvalscore estimator

XEvaluate a score by crossvalidation

modelselectionlearningcurve estimator X

yLearning curve

modelselectionpermutationtestscore Evaluate the significance of a crossvalidated score with
permutations

modelselectionvalidationcurve estimator

Validation curve

sklearnmodelselection crossvalidate

sklearnmodelselection crossvalidate estimator XyNone groupsNone scor
ingNone cv'warn' njobsNone verbose0

fitparamsNone predispatch'2njobs' re

turntrainscoreFalse returnestimatorFalse

errorscore'raisedeprecating'

Evaluate metrics by crossvalidation and also record fitscore times

Read more in the User Guide

Parameters

estimator estimator object implementing 'fit' The object to use to fit the data

Xarraylike The data to fit Can be for example a list or an array

yarraylike optional default None The target variable to try to predict in the case of super
vised learning

groups arraylike with shape nsamples optional Group labels for the samples used while
splitting the dataset into traintest set Only used in conjunction with a "Group" cvinstance

egGroupKFold

scoring string callable listtuple dict or None default None A single string see The scoring
parameter defining model evaluation rules or a callable see Defining your scoring strategy
from metric functions to evaluate the predictions on the test set

For evaluating multiple metrics either give a list of unique strings or a dict with names as
keys and callables as values

NOTE that when using custom scorers each scorer should return a single value Metric
functions returning a listarray of values can be wrapped into multiple scorers that return
one value each

SeeSpecifying multiple metrics for evaluation for an example

If None the estimator's score method is used

cvint crossvalidation generator or an iterable optional Determines the crossvalidation split
ting strategy Possible inputs for cv are

- None to use the default 3fold cross validation
- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter

626sklearnmodelselection Model Selection 2111

scikitlearn user guide Release 0213

- An iterable yielding train test splits as arrays of indices
- For integerNone inputs if the estimator is a classifier and yis either binary or multiclass StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

njobs int or None optional defaultNone The number of CPUs to use to do the computa tionNone means 1 unless in a joblibparallelbackend context1means using

all processors See Glossary for more details

verbose integer optional The verbosity level

fitparams dict optional Parameters to pass to the fit method of the estimator

predispatch int or string optional Controls the number of jobs that get dispatched during

parallel execution Reducing this number can be useful to avoid an explosion of memory

consumption when more jobs get dispatched than CPUs can process This parameter can

be

- None in which case all the jobs are immediately created and spawned Use this for lightweight and fastrunning jobs to avoid delays due to ondemand spawning of the jobs
- An int giving the exact number of total jobs that are spawned
- A string giving an expression as a function of njobs as in ‘2njobs’

returntrainscore boolean defaultFalse Whether to include train scores Computing train

ing scores is used to get insights on how different parameter settings impact the overfit

tingunderfitting tradeoff However computing the scores on the training set can be com

putationally expensive and is not strictly required to select the parameters that yield the best

generalization performance

returnestimator boolean default False Whether to return the estimators fitted on each split

errorscore ‘raise’ ‘raisedeprecating’ or numeric Value to assign to the score if an error

occurs in estimator fitting If set to ‘raise’ the error is raised If set to ‘raisedeprecating’

a FutureWarning is printed before the error is raised If a numeric value is given FitFailed

Warning is raised This parameter does not affect the refit step which will always raise the

error Default is ‘raisedeprecating’ but from version 022 it will change to nphan

Returns

scores dict of float arrays of shapensplits Array of scores of the estimator for each run of

the cross validation

A dict of arrays containing the scoretime arrays for each scorer is returned The possible

keys for this dict are

testscore The score array for test scores on each cv split

trainscore The score array for train scores on each cv split This is available

only ifreturntrainscore parameter is True

fittime The time for fitting the estimator on the train set for each cv split

scoretime The time for scoring the estimator on the test set for each cv

split Note time for scoring on the train set is not included even if

returntrainscore is set toTrue

2112 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

estimator The estimator objects for each cv split This is available only if
returnestimator parameter is set to True

See also

sklearnmodelselectioncrossvalscore Run crossvalidation for single metric evaluation

sklearnmodelselectioncrossvalpredict Get predictions from each split of cross

validation for diagnostic purposes

sklearnmetricsmakescorer Make a scorer from a performance metric or loss function

Examples

```
from sklearn import datasets linearmodel
from sklearnmodelselection import crossvalidate
from sklearnmetricsscorer import makescorer
from sklearnmetrics import confusionmatrix
from sklearnsvm import LinearSVC
```

```
diabetes datasetsloaddiabetes
```

```
X diabetesdata150
```

```
y diabetestarget150
```

```
lasso linearmodelLasso
```

Single metric evaluation using crossvalidate

```
cvresults crossvalidatelasso X y cv3
```

```
sortedcvresultskeys
```

```
fittime scoretime testscore
```

```
cvresultstestscore
```

```
array033150734 008022311 003531764
```

Multiple metric evaluation using crossvalidate please refer the scoring parameter doc for more in
formation

```
scores crossvalidatelasso X y cv3
```

```
scoringr2 negmeansquarederror
```

```
returntrainscore True
```

```
printscorestestnegmeansquarederror
```

```
36355 35733 61147
```

```
printscorestrainr2
```

```
028010158 039088426 022784852
```

```
sklearnmodelselection crossvalpredict
```

```
sklearnmodelselection crossvalpredict estimator XyNone groupsNone
```

```
cv'warn' njobsNone verbose0
```

```
fitparamsNone predispatch'2njobs'
```

```
method'predict'
```

Generate crossvalidated estimates for each input data point

The data is split according to the cv parameter Each sample belongs to exactly one test set and its prediction is
computed with an estimator fitted on the corresponding training set

Passing these predictions into an evaluation metric may not be a valid way to measure generalization perfor
mance Results can differ from crossvalidate andcrossvalscore unless all tests sets have equal

size and the metric decomposes over samples

626sklearnmodelselection Model Selection 2113

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

estimator estimator object implementing ‘fit’ and ‘predict’ The object to use to fit the data

Xarraylike The data to fit Can be for example a list or an array at least 2d

yarraylike optional default None The target variable to try to predict in the case of supervised learning

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into train/test set Only used in conjunction with a “Group” cvinstance eg GroupKFold

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold cross validation
- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs if the estimator is a classifier and y is either binary or multiclass

StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

njobs int or None optional defaultNone The number of CPUs to use to do the computationNone means 1 unless in a joblibparallelbackend context1means using

all processors See Glossary for more details

verbose integer optional The verbosity level

fitparams dict optional Parameters to pass to the fit method of the estimator

predispach int or string optional Controls the number of jobs that get dispatched during parallel execution Reducing this number can be useful to avoid an explosion of memory consumption when more jobs get dispatched than CPUs can process This parameter can be

- None in which case all the jobs are immediately created and spawned Use this for lightweight and fast-running jobs to avoid delays due to on-demand spawning of the jobs
- An int giving the exact number of total jobs that are spawned
- A string giving an expression as a function of njobs as in ‘2njobs’

method string optional default ‘predict’ Invokes the passed method name of the passed estimator For method ‘predict_proba’ the columns correspond to the classes in sorted order

Returns

predictions ndarray This is the result of calling method

See also

crossvalscore calculate score for each CV (Cross-Validation) split

2114 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

crossvalidate calculate one or more scores and timings for each CV (Cross-Validation) split

Notes

In the case that one or more classes are absent in a training portion a default score needs to be assigned to all instances for that class if method produces columns per class as in ‘decisionfunction’ ‘predictproba’ ‘predictlogproba’ For predictproba this value is 0 In order to ensure finite output we approximate negative infinity by the minimum finite float value for the dtype in other cases

Examples

```
from sklearn import datasets linearmodel
from sklearnmodelselection import crossvalpredict
diabetes datasetsloaddiabetes
X diabetesdata150
y diabetestarget150
lasso linearmodelLasso
ypred crossvalpredictlasso X y cv3
```

Examples using sklearnmodelselectioncrossvalpredict

```
•Plotting CrossValidated Predictions
sklearnmodelselection crossvalscore
sklearnmodelselection crossvalscore estimator XyNone groupsNone scor
ingNone cv‘warn’ njobsNone verbose0
fitparamsNone predispatch‘2njobs’
errorscore‘raisedeprecating’
```

Evaluate a score by crossvalidation

Read more in the User Guide

Parameters

estimator estimator object implementing ‘fit’ The object to use to fit the data
Xarraylike The data to fit Can be for example a list or an array
yarraylike optional default None The target variable to try to predict in the case of super
vised learning
groups arraylike with shape nsamples optional Group labels for the samples used while
splitting the dataset into traintest set Only used in conjunction with a “Group” cvinstance
egGroupKFold
scoring string callable or None optional default None A string see model evaluation doc
umentation or a scorer callable object function with signature scorere estimator
X y which should return only a single value
Similar tocrossvalidate but only a single metric is permitted
If None the estimator’s default scorer if available is used
626sklearnmodelselection Model Selection 2115

scikitlearn user guide Release 0213

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold cross validation
- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs if the estimator is a classifier and yis either binary or multiclass StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

njobs int or None optional defaultNone The number of CPUs to use to do the computationNone means 1 unless in a joblibparallelbackend context1means using

all processors See Glossary for more details

verbose integer optional The verbosity level

fitparams dict optional Parameters to pass to the fit method of the estimator

predispach int or string optional Controls the number of jobs that get dispatched during parallel execution Reducing this number can be useful to avoid an explosion of memory consumption when more jobs get dispatched than CPUs can process This parameter can be

- None in which case all the jobs are immediately created and spawned Use this for lightweight and fastrunning jobs to avoid delays due to ondemand spawning of the jobs
- An int giving the exact number of total jobs that are spawned
- A string giving an expression as a function of njobs as in '2njobs'

errorscore 'raise' 'raisedeprecating' or numeric Value to assign to the score if an error occurs in estimator fitting If set to 'raise' the error is raised If set to 'raisedeprecating' a FutureWarning is printed before the error is raised If a numeric value is given FitFailed Warning is raised This parameter does not affect the refit step which will always raise the error Default is 'raisedeprecating' but from version 022 it will change to nphan

Returns

scores array of float shapelenlistcv Array of scores of the estimator for each run of the cross validation

See also

sklearnmodelselectioncrossvalidate To run crossvalidation on multiple metrics and also to return train scores fit times and score times

sklearnmodelselectioncrossvalpredict Get predictions from each split of cross validation for diagnostic purposes

sklearnmetricsmakescorer Make a scorer from a performance metric or loss function

2116 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

```
from sklearn import datasets linearmodel
from sklearnmodelselection import crossvalscore
diabetes datasetsloadaddiabetes
X diabetesdata150
y diabetestarget150
lasso linearmodelLasso
printcrossvalscorelasso X y cv3
033150734 008022311 003531764
```

Examples using sklearnmodelselectioncrossvalscore

- Model selection with Probabilistic PCA (Principal Component Analysis) and Factor Analysis FA
- Crossvalidation on Digits Dataset Exercise
- Imputing missing values with variants of IterativeImputer
- Imputing missing values before building an estimator
- Underfitting vs Overfitting
- Nested versus nonnested crossvalidation
- SVMAnova SVM (Support Vector Machine) with univariate feature selection

```
sklearnmodelselection learningcurve
sklearnmodelselection learningcurve estimator X y groupsNone
trainsizesarray01 0325 0550775
1 cv'warn' scoringNone ex
ploitincrementalllearningFalse njobsNone
predispach'all' verbose0 shuffleFalse ran
domstateNone errorscore'raisedeprecating'
Learning curve
```

Determines crossvalidated training and test scores for different training set sizes

A crossvalidation generator splits the whole dataset k times in training and test data Subsets of the training set with varying sizes will be used to train the estimator and a score for each training subset size and the test set will be computed Afterwards the scores will be averaged over all k runs for each training subset size

Read more in the User Guide

Parameters

estimator object type that implements the “fit” and “predict” methods An object of that type which is cloned for each validation

Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples or nsamples nfeatures optional Target relative to X for classification or regression None for unsupervised learning

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set Only used in conjunction with a “Group” cvinstance

egGroupKFold

scikitlearn user guide Release 0213

trainsizes arraylike shape n ticks dtype float or int Relative or absolute numbers of training examples that will be used to generate the learning curve If the dtype is float it is regarded as a fraction of the maximum size of the training set that is determined by the selected validation method ie it has to be within 0 1 Otherwise it is interpreted as absolute sizes of the training sets Note that for classification the number of samples usually have to be big enough to contain at least one sample from each class default nplinspace01 10 5

cv int crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold cross validation
- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integer None inputs if the estimator is a classifier and y is either binary or multiclass StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cv default value if None will change from 3fold to 5fold in v022

scoring string callable or None optional default None A string see model evaluation documentation or a scorer callable object function with signature scorer(estimator

X y

exploit incremental learning boolean optional default False If the estimator supports incremental learning this will be used to speed up fitting for different training set sizes

njobs int or None optional default None Number of jobs to run in parallel None means 1 unless in a joblibparallelbackend context 1 means using all processors See

Glossary for more details

pre_dispatch integer or string optional Number of pre-dispatched jobs for parallel execution default is all The option can reduce the allocated memory The string can be an expression like '2*njobs'

verbose integer optional Controls the verbosity the higher the more messages

shuffle boolean optional Whether to shuffle training data before taking prefixes of it based on 'trainsizes'

random_state int RandomState instance or None optional default None If int ran

dom_state is the seed used by the random number generator If RandomState instance

random_state is the random number generator If None the random number generator is

the RandomState instance used by nprandom Used when shuffle is True

error_score 'raise' 'raisedeprecating' or numeric Value to assign to the score if an error

occurs in estimator fitting If set to 'raise' the error is raised If set to 'raisedeprecating'

a FutureWarning is printed before the error is raised If a numeric value is given FitFailed

Warning is raised This parameter does not affect the refit step which will always raise the

error Default is 'raisedeprecating' but from version 022 it will change to npran

Returns

trainsizes abs array shape n unique ticks dtype int Numbers of training examples that

has been used to generate the learning curve Note that the number of ticks might be less than n_ticks because duplicate entries will be removed

2118 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
trainscores array shape nticks ncvfolds Scores on training sets
testscores array shape nticks ncvfolds Scores on test set
Notes
Seeexamplesmodelselectionplotlearningcurvepy
Examples using sklearnmodelselectionlearningcurve
•Comparison of kernel ridge regression and SVR (Support Vector Regression)
•Plotting Learning Curves
sklearnmodelselection permutationtestscore
sklearnmodelselection permutationtestscore estimator Xy groupsNone
cv'warn' npermutations100
njobsNone randomstate0 ver
bose0 scoringNone
Evaluate the significance of a crossvalidated score with permutations
Read more in the User Guide
Parameters
estimator estimator object implementing 'fit' The object to use to fit the data
Xarraylike of shape at least 2D The data to fit
yarraylike The target variable to try to predict in the case of supervised learning
groups arraylike with shape nsamples optional Labels to constrain permutation within
groups ie yvalues are permuted among samples with the same group identifier When not
specifiedyvalues are permuted among all samples
When a grouped crossvalidator is used the group labels are also passed on to the split
method of the crossvalidator The crossvalidator uses them for grouping the samples while
splitting the dataset into traintest set
scoring string callable or None optional default None A single string see The scoring
parameter defining model evaluation rules or a callable see Defining your scoring strategy
from metric functions to evaluate the predictions on the test set
If None the estimator's score method is used
cvint crossvalidation generator or an iterable optional Determines the crossvalidation split
ting strategy Possible inputs for cv are
• None to use the default 3fold cross validation
• integer to specify the number of folds in a StratifiedKfold
• CV (Cross-Validation) splitter
• An iterable yielding train test splits as arrays of indices
626sklearnmodelselection Model Selection 2119

scikitlearn user guide Release 0213

For integerNone inputs if the estimator is a classifier and yis either binary or multiclass StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

npermutations integer optional Number of times to permute y

njobs int or None optional defaultNone The number of CPUs to use to do the computationNone means 1 unless in a joblibparallelbackend context1means using

all processors See Glossary for more details

randomstate int RandomState instance or None optional default0 If int randomstate

is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState

instance used by nprandom

verbose integer optional The verbosity level

Returns

score float The true score without permuting targets

permutationscores array shape npermutations The scores obtained for each permutation

pvalue float The pvalue which approximates the probability that the score would be obtained by chance This is calculated as

$C - 1$ / $npermutations - 1$

Where C is the number of permutations whose score \geq the true score

The best possible pvalue is 1/npermutations - 1 the worst is 10

Notes

This function implements Test 1 in

Ojala and Garriga Permutation Tests for Studying Classifier Performance The Journal of Machine Learning Research 2010 vol 11

Examples using sklearnmodelselectionpermutationtestscore

•Test with permutations the significance of a classification score

sklearnmodelselection validationcurve

sklearnmodelselection validationcurve estimator Xyparamname paramrange

groupsNone cv'warn' scoringNone

njobsNone predispach'all' verbose0

errorscore'raisedeprecating'

Validation curve

Determine training and test scores for varying parameter values

2120 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Compute scores for an estimator with different values of a specified parameter This is similar to grid search with one parameter However this will also compute training scores and is merely a utility for plotting the results

Read more in the User Guide

Parameters

estimator object type that implements the “fit” and “predict” methods An object of that type which is cloned for each validation

Xarraylike shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples or nsamples nfeatures optional Target relative to X for classification or regression None for unsupervised learning

paramname string Name of the parameter that will be varied

paramrange arraylike shape nvalues The values of the parameter that will be evaluated

groups arraylike with shape nsamples optional Group labels for the samples used while splitting the dataset into traintest set Only used in conjunction with a “Group” cvinstance

egGroupKFold

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold cross validation
- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs if the estimator is a classifier and yis either binary or multiclass

StratifiedKFold is used In all other cases KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

scoring string callable or None optional default None A string see model evaluation documentation or a scorer callable object function with signature scorerestimator

X y

njobs int or None optional defaultNone Number of jobs to run in parallel None means 1 unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

predispatch integer or string optional Number of predispatched jobs for parallel execution default is all The option can reduce the allocated memory The string can be an expression like ‘2njobs’

verbose integer optional Controls the verbosity the higher the more messages

errorscore ‘raise’ ‘raisedeprecating’ or numeric Value to assign to the score if an error occurs in estimator fitting If set to ‘raise’ the error is raised If set to ‘raisedeprecating’

a FutureWarning is printed before the error is raised If a numeric value is given FitFailed Warning is raised This parameter does not affect the refit step which will always raise the error Default is ‘raisedeprecating’ but from version 022 it will change to nphan

Returns

626sklearnmodelselection Model Selection 2121

scikitlearn user guide Release 0213
trainscores array shape nticks ncvfolds Scores on training sets
testscores array shape nticks ncvfolds Scores on test set
Notes

SeePlotting Validation Curves
Examples using sklearnmodelselectionvalidationcurve

•Plotting Validation Curves
627sklearnmulticlass Multiclass and multilabel classification

6271 Multiclass and multilabel classification strategies

This module implements multiclass learning algorithms

- onevstherest onevsall
- onevsone
- error correcting output codes

The estimators provided in this module are metaestimators they require a base estimator to be provided in their constructor For example it is possible to use these estimators to turn a binary classifier or a regressor into a multiclass classifier It is also possible to use these estimators with multiclass estimators in the hope that their accuracy or runtime performance improves

All classifiers in scikitlearn implement multiclass classification you only need to use this module if you want to experiment with custom multiclass strategies

The onevstherest metaclassifier also implements a predictproba method so long as such a method is implemented by the base classifier This method returns probabilities of class membership in both the single label and multilabel case Note that in the multilabel case probabilities are the marginal probability that a given sample falls in the given class As such in the multilabel case the sum of these probabilities over all possible labels for a given sample will not sum to unity as they do in the single label case

User guide See the Multiclass and multilabel algorithms section for further details

multiclassOneVsRestClassifier estimator

Onevstherest OvR multiclassmultilabel strategy

multiclassOneVsOneClassifier estimator

Onevsone multiclass strategy

multiclassOutputCodeClassifier estimator

ErrorCorrecting OutputCode multiclass strategy

6272sklearnmulticlass OneVsRestClassifier

classsklearnmulticlass OneVsRestClassifier estimator njobsNone

Onevstherest OvR multiclassmultilabel strategy

Also known as onevsall this strategy consists in fitting one classifier per class For each classifier the class is

2122 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

fitted against all the other classes In addition to its computational efficiency only nclasses classifiers are needed one advantage of this approach is its interpretability Since each class is represented by one and one classifier only it is possible to gain knowledge about the class by inspecting its corresponding classifier This is the most commonly used strategy for multiclass classification and is a fair default choice This strategy can also be used for multilabel learning where a classifier is used to predict multiple labels for instance by fitting on a 2d matrix in which cell i j is 1 if sample i has label j and 0 otherwise In the multilabel learning literature OvR is also known as the binary relevance method

Read more in the User Guide

Parameters

estimator estimator object An estimator object implementing fit and one of decisionfunction or predict_proba
njobs int or None optional default None The number of jobs to use for the computation
None means 1 unless in a joblibparallelbackend context 1 means using all processors See Glossary for more details

Attributes

estimators list of nclasses estimators Estimators used for predictions
classes array shape (nclasses) Class labels
labelbinarizer LabelBinarizer object Object used to transform multiclass labels to binary labels and viceversa
multilabel boolean Whether this is a multilabel classifier

Methods

decisionfunction self X Returns the distance of each sample from the decision boundary for each class
fitself X y Fit underlying estimators
getparams self deep Get parameters for this estimator
partialfit self X y classes Partially fit underlying estimators
predict self X Predict multiclass targets using underlying estimators
predict_proba self X Probability estimates
score self X y sampleweight Returns the mean accuracy on the given test data and labels
setparams self params Set the parameters of this estimator
init self estimator njobs None
decisionfunction self X

Returns the distance of each sample from the decision boundary for each class This can only be used with estimators which implement the decisionfunction method

Parameters

X arraylike shape (nsamples, nfeatures)

Returns

T arraylike shape (nsamples, nclasses)

fitself X y

627 sklearnmulticlass Multiclass and multilabel classification 2123

scikitlearn user guide Release 0213

Fit underlying estimators

Parameters

Xsparse arraylike shape nsamples nfeatures Data

ysparse arraylike shape nsamples nsamples nclasses Multiclass targets

An indicator matrix turns on multilabel classification

Returns

self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

multilabel

Whether this is a multilabel classifier

partialfit selfXyclassesNone

Partially fit underlying estimators

Should be used when memory is inefficient to train all data Chunks of data can be passed in several

iteration

Parameters

Xsparse arraylike shape nsamples nfeatures Data

ysparse arraylike shape nsamples nsamples nclasses Multiclass targets

An indicator matrix turns on multilabel classification

classes array shape nclasses Classes across all calls to partialfit Can be obtained

vianpuniqueyall where yall is the target vector of the entire dataset This

argument is only required in the first call of partialfit and can be omitted in the subsequent

calls

Returns

self

predictselfX

Predict multiclass targets using underlying estimators

Parameters

Xsparse arraylike shape nsamples nfeatures Data

Returns

ysparse arraylike shape nsamples nsamples nclasses Predicted multiclass

targets

predictproba selfX

Probability estimates

The returned estimates for all classes are ordered by label of classes

2124 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Note that in the multilabel case each sample can have any number of labels This returns the marginal probability that the given sample has the label in question For example it is entirely consistent that two labels both have a 90 probability of applying to a given sample

In the single label multiclass case the rows of the returned matrix sum to 1

Parameters

Xarraylike shape nsamples nfeatures

Returns

Tsparse arraylike shape nsamples nclasses Returns the probability of the sample for each class in the model where classes are ordered as they are in selfclasses

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnmulticlassOneVsRestClassifier

- Multilabel classification
- Receiver Operating Characteristic ROC
- PrecisionRecall
- Classifier Chain

6273sklearnmulticlass OneVsOneClassifier

classssklearnmulticlass OneVsOneClassifier estimator njobsNone

Onevsone multiclass strategy

This strategy consists in fitting one classifier per class pair At prediction time the class which received the most votes is selected Since it requires to fit nclasses nclasses 1 2 classifiers this method

is usually slower than onevstherest due to its Onclasses2 complexity However this method may be advantageous for algorithms such as kernel algorithms which don't scale well with nsamples This is because 6273sklearnmulticlass Multiclass and multilabel classification 2125

scikitlearn user guide Release 0213

each individual learning problem only involves a small subset of the data whereas with onevs therest the complete dataset is used nclasses times

Read more in the User Guide

Parameters

estimator estimator object An estimator object implementing fit and one of decisionfunction or predict_proba

njobs int or None optional default None The number of jobs to use for the computation

None means 1 unless in a joblibparallelbackend context 1 means using all processors See Glossary for more details

Attributes

estimators list of nclasses nclasses 1 2 estimators Estimators used for predictions

classes numpy array of shape nclasses Array containing labels

pairwiseindices list length len(estimators) or None Indices of samples used when training the estimators None when estimator does not have pairwise attribute

Methods

decisionfunction self X Decision function for the OneVsOneClassifier

fitself X y Fit underlying estimators

getparams self deep Get parameters for this estimator

partialfit self X y classes Partially fit underlying estimators

predict self X Estimate the best class label for each sample in X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init self estimator njobs None

decisionfunction self X

Decision function for the OneVsOneClassifier

The decision values for the samples are computed by adding the normalized sum of pairwise classification confidence levels to the votes in order to disambiguate between the decision values when the votes for all the classes are equal leading to a tie

Parameters

Xarraylike shape nsamples nfeatures

Returns

Yarraylike shape nsamples nclasses

fitself X y

Fit underlying estimators

Parameters

Xsparse arraylike shape nsamples nfeatures Data

yarraylike shape nsamples Multiclass targets

2126 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

self

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyclassesNone

Partially fit underlying estimators

Should be used when memory is inefficient to train all data Chunks of data can be passed in several

iteration where the first call should have an array of all target variables

Parameters

Xsparse arraylike shape nsamples nfeatures Data

yarraylike shape nsamples Multiclass targets

classes array shape nclasses Classes across all calls to partialfit Can be obtained

via np.unique(yall) where yall is the target vector of the entire dataset This

argument is only required in the first call of partialfit and can be omitted in the subsequent

calls

Returns

self

predictselfX

Estimate the best class label for each sample in X

This is implemented as argmax(decision_function(X, axis=1)) which will return the label

of the class with most votes by estimators predicting the outcome of a decision for each possible class pair

Parameters

Xsparse arraylike shape nsamples nfeatures Data

Returns

numpy array of shape nsamples Predicted multiclass targets

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each

sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

627sklearnmulticlass Multiclass and multilabel classification 2127

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

6274sklearnmulticlass OutputCodeClassifier

classsklearnmulticlass OutputCodeClassifier estimator codesize15 ran

domstateNone njobsNone

ErrorCorrecting OutputCode multiclass strategy

Outputcode based strategies consist in representing each class with a binary code an array of 0s and 1s At fitting time one binary classifier per bit in the code book is fitted At prediction time the classifiers are used to project new points in the class space and the class closest to the points is chosen The main advantage of these strategies is that the number of classifiers used can be controlled by the user either for compressing the model 0 codesize 1 or for making the model more robust to errors codesize 1 See the documentation for

more details

Read more in the User Guide

Parameters

estimator estimator object An estimator object implementing fit and one of

decisionfunction orpredictproba

codesize float Percentage of the number of classes to be used to create the code book A number between 0 and 1 will require fewer classifiers than onevstherest A number greater than 1 will require more classifiers than onevstherest

randomstate int RandomState instance or None optional default None The generator used to initialize the codebook If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

njobs int or None optional defaultNone The number of jobs to use for the computation

None means 1 unless in a joblibparallelbackend context1means using all

processors See Glossary for more details

Attributes

estimators list ofintnclasses codesize estimators Estimators used for pre

dictions

classes numpy array of shape nclasses Array containing labels

codebook numpy array of shape nclasses codesize Binary array containing the code of each class

References

R2eddaaec08491 R2eddaaec08492 R2eddaaec08493

2128 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Methods

fitself X y Fit underlying estimators
getparams self deep Get parameters for this estimator
predict self X Predict multiclass targets using underlying estimators
score self X y sampleweight Returns the mean accuracy on the given test data and labels
setparams self params Set the parameters of this estimator
init selfestimator codesize15 randomstateNone njobsNone
fitselfXy
Fit underlying estimators
Parameters
Xsparse arraylike shape nsamples nfeatures Data
ynumpy array of shape nsamples Multiclass targets
Returns
self
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values
predictselfX
Predict multiclass targets using underlying estimators
Parameters
Xsparse arraylike shape nsamples nfeatures Data
Returns
ynumpy array of shape nsamples Predicted multiclass targets
scoreselfXysampleweightNone
Returns the mean accuracy on the given test data and labels
In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted
Parameters
Xarraylike shape nsamples nfeatures Test samples
yarraylike shape nsamples or nsamples noutputs True labels for X
sampleweight arraylike shape nsamples optional Sample weights
Returns
score float Mean accuracy of selfpredictX wrt y
627sklearnmulticlass Multiclass and multilabel classification 2129

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

628sklearnmultioutput Multioutput regression and classifica
tion

This module implements multioutput regression and classification

The estimators provided in this module are metaestimators they require a base estimator to be provided in their constructor The metaestimator extends single output estimators to multioutput estimators

User guide See the Multiclass and multilabel algorithms section for further details

multioutputClassifierChain baseestimator A multilabel model that arranges binary classifiers into a chain

multioutputMultiOutputRegressor estimator Multi target regression

multioutputMultiOutputClassifier estimator Multi target classification

multioutputRegressorChain baseestimator

A multilabel model that arranges regressions into a chain

6281sklearnmultioutput ClassifierChain

classsklearnmultioutput ClassifierChain baseestimator orderNone cvNone ran
domstateNone

A multilabel model that arranges binary classifiers into a chain

Each model makes a prediction in the order specified by the chain using all of the available features provided to the model plus the predictions of models that are earlier in the chain

Read more in the User Guide

Parameters

baseestimator estimator The base estimator from which the classifier chain is built

order arraylike shapenoutputs or ‘random’ optional By default the order will be deter
mined by the order of columns in the label matrix Y

order 0 1 2 Yshape1 1

The order of the chain can be explicitly set by providing a list of integers For example for
a chain of length 5

order 1 3 2 4 0

means that the first model in the chain will make predictions for column 1 in the Y matrix
the second model will make predictions for column 3 etc

2130 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

If order is 'random' a random ordering will be used
cvint crossvalidation generator or an iterable optional defaultNone Determines whether to use cross validated predictions or true labels for the results of previous estimators in the chain If cv is None the true labels are used when fitting Otherwise possible inputs for cv are

- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

randomstate int RandomState instance or None optional defaultNone If int ran
domstate is the seed used by the random number generator If RandomState instance
randomstate is the random number generator If None the random number generator is
the RandomState instance used by nprandom

The random number generator is used to generate random chain orders

Attributes

classes list A list of arrays of length lenestimators containing the class labels for
each estimator in the chain

estimators list A list of clones of baseestimator

order list The order of labels in the classifier chain

See also

RegressorChain Equivalent for regression

MultioutputClassifier Classifies each output independently rather than chaining

References

Jesse Read Bernhard Pfahringer Geoff Holmes Eibe Frank "Classifier Chains for Multilabel Classification"
2009

Methods

decisionfunction self X Evaluate the decisionfunction of the models in the
chain

fitself X Y Fit the model to data matrix X and targets Y

getparams self deep Get parameters for this estimator

predict self X Predict on the data matrix X using the ClassifierChain

model

predictproba self X Predict probability estimates

score self X y sampleweight Returns the mean accuracy on the given test data and
labels

setparams self params Set the parameters of this estimator

init selfbaseestimator orderNone cvNone randomstateNone

decisionfunction selfX

Evaluate the decisionfunction of the models in the chain

628sklearnmultioutput Multioutput regression and classification 2131

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures

Returns

Ydecision arraylike shape nsamples nclasses Returns the decision function of the sample for each model in the chain

fitselfXY

Fit the model to data matrix X and targets Y

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

Yarraylike shape nsamples nclasses The target values

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict on the data matrix X using the ClassifierChain model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

Returns

Ypred arraylike shape nsamples nclasses The predicted values

predictproba selfX

Predict probability estimates

Parameters

Xarraylike sparse matrix shape nsamples nfeatures

Returns

Yprob arraylike shape nsamples nclasses

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

2132 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnmultioutputClassifierChain

•Classifier Chain

6282sklearnmultioutput MultiOutputRegressor

classsklearnmultioutput MultiOutputRegressor estimator njobsNone

Multi target regression

This strategy consists of fitting one regressor per target This is a simple strategy for extending regressors that do not natively support multitarget regression

Parameters

estimator estimator object An estimator object implementing fit andpredict

njobs int or None optional defaultNone The number of jobs to run in parallel for fit

None means 1 unless in a joblibparallelbackend context1means using all

processors See Glossary for more details

When individual estimators are fast to train or predict using njobs1 can result in slower performance due to the overhead of spawning processes

Attributes

estimators list ofnoutput estimators Estimators used for predictions

Methods

fitself X y sampleweight Fit the model to data

getparams self deep Get parameters for this estimator

partialfit self X y sampleweight Incrementally fit the model to data

predict self X Predict multioutput variable using a model trained for each target variable

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfestimator njobsNone

fitselfXysampleweightNone

Fit the model to data Fit a separate model for each output variable

628sklearnmultioutput Multioutput regression and classification 2133

Parameters

Xsparse arraylike shape nsamples nfeatures Data
ysparse arraylike shape nsamples noutputs Multioutput targets An indicator matrix turns on multilabel estimation
sampleweight arraylike shape nsamples or None Sample weights If None then samples are equally weighted Only supported if the underlying regressor supports sample weights

Returns

self object
getparams selfdeepTrue
Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values
partialfit selfXysampleweightNone
Incrementally fit the model to data Fit a separate model for each output variable

Parameters

Xsparse arraylike shape nsamples nfeatures Data
ysparse arraylike shape nsamples noutputs Multioutput targets
sampleweight arraylike shape nsamples or None Sample weights If None then samples are equally weighted Only supported if the underlying regressor supports sample weights

Returns

self object
predictselfX
Predict multioutput variable using a model trained for each target variable

Parameters

Xsparse arraylike shape nsamples nfeatures Data
Returns

ysparse arraylike shape nsamples noutputs Multioutput targets predicted across multiple predictors Note Separate models are generated for each predictor
scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$ 2sum and v is the regression sum of squares $y_{true} - y_{true\text{mean}}$ 2sum Best possible score is 10 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

R2 is calculated by weighting all the targets equally using multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnmultioutputMultiOutputRegressor

•Comparing random forests and the multioutput meta estimator

6283sklearnmultioutput MultiOutputClassifier

classsklearnmultioutput MultiOutputClassifier estimator njobsNone

Multi target classification

This strategy consists of fitting one classifier per target This is a simple strategy for extending classifiers that

do not natively support multitarget classification

Parameters

estimator estimator object An estimator object implementing fitscore and

predictproba

njobs int or None optional defaultNone The number of jobs to use for the computa

tion It does each target variable in y in parallel None means 1 unless in a joblib

parallelbackend context1means using all processors See Glossary for more

details

Attributes

estimators list ofnoutput estimators Estimators used for predictions

Methods

fitself X y sampleweight Fit the model to data

getparams self deep Get parameters for this estimator

partialfit self X y classes sampleweight Incrementally fit the model to data

Continued on next page

628sklearnmultioutput Multioutput regression and classification 2135

predict self X Predict multioutput variable using a model trained for each target variable

predictproba self X Probability estimates

score self X y Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init selfestimator njobsNone

fitselfXysampleweightNone

Fit the model to data Fit a separate model for each output variable

Parameters

Xsparse arraylike shape nsamples nfeatures Data

ysparse arraylike shape nsamples noutputs Multioutput targets An indicator ma

trix turns on multilabel estimation

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Only supported if the underlying regressor supports sample weights

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyclassesNone sampleweightNone

Incrementally fit the model to data Fit a separate model for each output variable

Parameters

Xsparse arraylike shape nsamples nfeatures Data

ysparse arraylike shape nsamples noutputs Multioutput targets

classes list of numpy arrays shape noutputs Each array is unique classes for one output

in strint Can be obtained by via npuniquey i for i in rangey

shape1 where y is the target matrix of the entire dataset This argument is required

for the first call to partialfit and can be omitted in the subsequent calls Note that y doesn't need to contain all labels in classes

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Only supported if the underlying regressor supports sample weights

Returns

self object

predictselfX

scikitlearn user guide Release 0213

Predict multioutput variable using a model trained for each target variable

Parameters

Xsparse arraylike shape nsamples nfeatures Data

Returns

ysparse arraylike shape nsamples noutputs Multioutput targets predicted across

multiple predictors Note Separate models are generated for each predictor

predictproba selfX

Probability estimates Returns prediction probabilities for each class of each output

This method will raise a ValueError if any of the estimators do not have predictproba

Parameters

Xarraylike shape nsamples nfeatures Data

Returns

parray of shape nsamples nclasses or a list of noutputs such arrays if noutputs

1 The class probabilities of the input samples The order of the classes corresponds to

that in the attribute classes

scoresselfXy

Returns the mean accuracy on the given test data and labels

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples noutputs True values for X

Returns

scores float accuracyscore of selfpredictX versus y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

6284sklearnmultioutput RegressorChain

classsklearnmultioutput RegressorChain baseestimator orderNone cvNone ran

domstateNone

A multilabel model that arranges regressions into a chain

Each model makes a prediction in the order specified by the chain using all of the available features provided to

the model plus the predictions of models that are earlier in the chain

Read more in the User Guide

Parameters

baseestimator estimator The base estimator from which the classifier chain is built

628sklearnmultioutput Multioutput regression and classification 2137

scikitlearn user guide Release 0.21.3

order arraylike shapenoutputs or 'random' optional By default the order will be determined by the order of columns in the label matrix Y

order 0 1 2 Yshape1 1

The order of the chain can be explicitly set by providing a list of integers For example for a chain of length 5

order 1 3 2 4 0

means that the first model in the chain will make predictions for column 1 in the Y matrix the second model will make predictions for column 3 etc

If order is 'random' a random ordering will be used

cvint crossvalidation generator or an iterable optional defaultNone Determines whether to use cross validated predictions or true labels for the results of previous estimators in the chain If cv is None the true labels are used when fitting Otherwise possible inputs for cv are

- integer to specify the number of folds in a StratifiedKFold
- CV (Cross-Validation) splitter

• An iterable yielding train test splits as arrays of indices

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

The random number generator is used to generate random chain orders

Attributes

estimators list A list of clones of baseestimator

order list The order of labels in the classifier chain

See also

ClassifierChain Equivalent for classification

MultioutputRegressor Learns each output independently rather than chaining

Methods

fitself X Y Fit the model to data matrix X and targets Y

getparams self deep Get parameters for this estimator

predict self X Predict on the data matrix X using the ClassifierChain

model

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfbaseestimator orderNone cvNone randomstateNone

fitselfXY

Fit the model to data matrix X and targets Y

2138 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

Yarraylike shape nsamples nclasses The target values

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict on the data matrix X using the ClassifierChain model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

Returns

Ypred arraylike shape nsamples nclasses The predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

628sklearnmultioutput Multioutput regression and classification 2139

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

629sklearnnaivebayes Naive Bayes

Thesklearnnaivebayes module implements Naive Bayes algorithms These are supervised learning methods based on applying Bayes’ theorem with strong naive feature independence assumptions

User guide See the Naive Bayes section for further details

naivebayesBernoulliNB alpha binarize Naive Bayes classifier for multivariate Bernoulli models

naivebayesGaussianNB priors varsmoothing Gaussian Naive Bayes GaussianNB

naivebayesMultinomialNB alpha Naive Bayes classifier for multinomial models

naivebayesComplementNB alpha fitprior The Complement Naive Bayes classifier described in Ren nie et al

6291sklearnnaivebayes BernoulliNB

classsklearnnaivebayes BernoulliNB alpha10 binarize00 fitpriorTrue

classpriorNone

Naive Bayes classifier for multivariate Bernoulli models

Like MultinomialNB this classifier is suitable for discrete data The difference is that while MultinomialNB works with occurrence counts BernoulliNB is designed for binaryboolean features

Read more in the User Guide

Parameters

alpha float optional default10 Additive LaplaceLidstone smoothing parameter 0 for no smoothing

binarize float or None optional default00 Threshold for binarizing mapping to booleans

of sample features If None input is presumed to already consist of binary vectors

fitprior boolean optional defaultTrue Whether to learn class prior probabilities or not If false a uniform prior will be used

classprior arraylike sizenclasses optional defaultNone Prior probabilities of the classes If specified the priors are not adjusted according to the data

Attributes

classlogprior array shape nclasses Log probability of each class smoothed

featurelogprob array shape nclasses nfeatures Empirical log probability of fea tures given a class Pxiy

classcount array shape nclasses Number of samples encountered for each class during fitting This value is weighted by the sample weight when provided

2140 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

featurecount array shape nclasses nfeatures Number of samples encountered for each class feature during fitting This value is weighted by the sample weight when provided

References

CD Manning P Raghavan and H Schuetze 2008 Introduction to Information Retrieval Cambridge University Press pp 234265 <https://nlp.stanford.edu/IR-book/html/htmledition/the-bernoulli-model-1.html>
A McCallum and K Nigam 1998 A comparison of event models for naive Bayes text classification Proc AAAI/CML98 Workshop on Learning for Text Categorization pp 4148
V Metsis I Androutsopoulos and G Paliouras 2006 Spam filtering with naive Bayes – Which naive Bayes 3rd Conf on Email and AntiSpam CEAS

Examples

```
import numpy as np
X = np.random.randint(2, size=(6, 100))
Y = np.array([2, 3, 4, 4, 5])
from sklearn.naive_bayes import BernoulliNB
clf = BernoulliNB()
clf.fit(X, Y)
BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)
print(clf.predict(X[2:3]))
```

Methods

fit(self, X, y, sample_weight) Fit Naive Bayes classifier according to X, y
get_params(self, deep=True) Get parameters for this estimator
partial_fit(self, X, y, classes, sample_weight) Incremental fit on a batch of samples
predict(self, X) Perform classification on an array of test vectors X
predict_log_proba(self, X) Return log-probability estimates for the test vector X
predict_proba(self, X) Return probability estimates for the test vector X
score(self, X, y, sample_weight) Returns the mean accuracy on the given test data and labels
set_params(self, **params) Set the parameters of this estimator
init(self, alpha=1.0, binarize=0.0, fit_prior=True, class_prior=None)
fit(self, X, y, sample_weight=None)
Fit Naive Bayes classifier according to X, y

Parameters

X array-like, sparse matrix shape (n_samples, n_features) Training vectors, where n_samples is the number of samples and n_features is the number of features
y array-like shape (n_samples) Target values
sample_weight array-like shape (n_samples) default=None Weights applied to individual samples (1 for unweighted)
629sklearn.naive_bayes Naive Bayes 2141

scikitlearn user guide Release 0213

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyclassesNone sampleweightNone

Incremental fit on a batch of samples

This method is expected to be called several times consecutively on different chunks of a dataset so as to

implement outofcore or online learning

This is especially useful when the whole dataset is too big to fit in memory at once

This method has some performance overhead hence it is better to call partialfit on chunks of data that are

as large as possible as long as fitting in the memory budget to hide the overhead

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where

nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target values

classes arraylike shape nclasses defaultNone List of all the classes that can pos

sibly appear in the y vector

Must be provided at the first call to partialfit can be omitted in subsequent calls

sampleweight arraylike shape nsamples defaultNone Weights applied to indi

vidual samples 1 for unweighted

Returns

self object

predictselfX

Perform classification on an array of test vectors X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carray shape nsamples Predicted target values for X

predictlogproba selfX

Return logprobability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

2142 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Carraylike shape nsamples nclasses Returns the logprobability of the samples for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

predictproba selfX

Return probability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carraylike shape nsamples nclasses Returns the probability of the samples for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnnaivebayesBernoulliNB

•Hashing feature transformation using Totally Random Trees

•Classification of text documents using sparse features

6292sklearnnaivebayes GaussianNB

classssklearnnaivebayes GaussianNB priorsNone varsmoothing1e09

Gaussian Naive Bayes GaussianNB

Can perform online updates to model parameters via partialfit method For details on algorithm used to update feature means and variance online see Stanford CS tech report STANCS79773 by Chan Golub and LeVeque

httpistanfordedupubcstrreportsctr79773CSTR79773pdf

629sklearnnaivebayes Naive Bayes 2143

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

priors arraylike shape nclasses Prior probabilities of the classes If specified the priors are not adjusted according to the data

varsmoothing float optional default1e9 Portion of the largest variance of all features that is added to variances for calculation stability

Attributes

classprior array shape nclasses probability of each class

classcount array shape nclasses number of training samples observed in each class

theta array shape nclasses nfeatures mean of each feature per class

sigma array shape nclasses nfeatures variance of each feature per class

epsilon float absolute additive value to variances

Examples

```
import numpy as np
X nparray1 1 2 1 3 2 1 1 2 1 3 2
Y nparray1 1 1 2 2 2
from sklearnnaivebayes import GaussianNB
clf GaussianNB
clffitX Y
GaussianNBpriorsNone varsmoothing1e09
printclfpredict08 1
1
clfpf GaussianNB
clfpfpartialfitX Y npuniqueY
GaussianNBpriorsNone varsmoothing1e09
printclfpfpredict08 1
1
```

Methods

fitself X y sampleweight Fit Gaussian Naive Bayes according to X y

getparams self deep Get parameters for this estimator

partialfit self X y classes sampleweight Incremental fit on a batch of samples

predict self X Perform classification on an array of test vectors X

predictlogproba self X Return logprobability estimates for the test vector X

predictproba self X Return probability estimates for the test vector X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init selfpriorsNone varsmoothing1e09

fitselfXysampleweightNone

Fit Gaussian Naive Bayes according to X y

Parameters

2144 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target values

sampleweight arraylike shape nsamples optional defaultNone Weights applied to individual samples 1 for unweighted

New in version 017 Gaussian Naive Bayes supports fitting with sampleweight

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyclassesNone sampleweightNone

Incremental fit on a batch of samples

This method is expected to be called several times consecutively on different chunks of a dataset so as to implement outofcore or online learning

This is especially useful when the whole dataset is too big to fit in memory at once

This method has some performance and numerical stability overhead hence it is better to call partialfit on chunks of data that are as large as possible as long as fitting in the memory budget to hide the overhead

Parameters

Xarraylike shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target values

classes arraylike shape nclasses optional defaultNone List of all the classes that can possibly appear in the y vector

Must be provided at the first call to partialfit can be omitted in subsequent calls

sampleweight arraylike shape nsamples optional defaultNone Weights applied to individual samples 1 for unweighted

New in version 017

Returns

self object

predictselfX

Perform classification on an array of test vectors X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carray shape nsamples Predicted target values for X

629sklearnnaivebayes Naive Bayes 2145

scikitlearn user guide Release 0213

predictlogproba selfX

Return logprobability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carraylike shape nsamples nclasses Returns the logprobability of the samples

for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

predictproba selfX

Return probability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carraylike shape nsamples nclasses Returns the probability of the samples for

each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnnaivebayesGaussianNB

- Comparison of Calibration of Classifiers
 - Probability Calibration curves
 - Probability calibration of classifiers
 - Classifier comparison
 - Plot class probabilities calculated by the VotingClassifier
- 2146 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- Plotting Learning Curves
- Importance of Feature Scaling

6293sklearnnaivebayes MultinomialNB

classsklearnnaivebayes MultinomialNB alpha10 fitpriorTrue classpriorNone

Naive Bayes classifier for multinomial models

The multinomial Naive Bayes classifier is suitable for classification with discrete features eg word counts for text classification The multinomial distribution normally requires integer feature counts However in practice fractional counts such as tfidf may also work

Read more in the User Guide

Parameters

alpha float optional default10 Additive LaplaceLidstone smoothing parameter 0 for no smoothing

fitprior boolean optional defaultTrue Whether to learn class prior probabilities or not If false a uniform prior will be used

classprior arraylike size nclasses optional defaultNone Prior probabilities of the classes If specified the priors are not adjusted according to the data

Attributes

classlogprior array shape nclasses Smoothed empirical log probability for each class

intercept array shape nclasses Mirrors classlogprior for interpreting MultinomialNB as a linear model

featurelogprob array shape nclasses nfeatures Empirical log probability of features given a class Pxiy

coef array shape nclasses nfeatures Mirrors featurelogprob for interpreting MultinomialNB as a linear model

classcount array shape nclasses Number of samples encountered for each class during fitting This value is weighted by the sample weight when provided

featurecount array shape nclasses nfeatures Number of samples encountered for each class feature during fitting This value is weighted by the sample weight when provided

Notes

For the rationale behind the names coef andintercept ie naive Bayes as a linear classifier see J Rennie et al 2003 Tackling the poor assumptions of naive Bayes text classifiers ICML

References

CD Manning P Raghavan and H Schuetze 2008 Introduction to Information Retrieval Cambridge University Press pp 234265 <https://nlp.stanford.edu/IRbook/html/html/editionnaivebayestextclassification1.html>

6293sklearnnaivebayes Naive Bayes 2147

scikitlearn user guide Release 0213

Examples

```
import numpy as np
X = np.random.randint(5, size=(6, 100))
y = np.array([1, 2, 3, 4, 5, 6])
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB()
clf.fit(X, y)
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
print(clf.predict(X[2:3]))
```

Methods

```
fit(self, X, y, sample_weight=None) Fit Naive Bayes classifier according to X, y
get_params(self, deep=True) Get parameters for this estimator
partial_fit(self, X, y, classes=None, sample_weight=None) Incremental fit on a batch of samples
predict(self, X) Perform classification on an array of test vectors X
predict_log_proba(self, X) Return log-probability estimates for the test vector X
predict_proba(self, X) Return probability estimates for the test vector X
score(self, X, y, sample_weight=None) Returns the mean accuracy on the given test data and labels
```

```
set_params(self, **kwargs) Set the parameters of this estimator
```

```
init(self, alpha=1.0, fit_prior=True, class_prior=None)
```

```
fit(self, X, y, sample_weight=None)
```

Fit Naive Bayes classifier according to X, y

Parameters

X: array-like, sparse matrix, shape = (n_samples, n_features) Training vectors, where n_samples is the number of samples and n_features is the number of features
y: array-like, shape = (n_samples,) Target values
sample_weight: array-like, shape = (n_samples,) Weights applied to individual samples (1 for unweighted)

Returns

```
self object
get_params(self, deep=True) Get parameters for this estimator
```

Parameters

deep: boolean, optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params: mapping of string to any Parameter names mapped to their values
partial_fit(self, X, y, classes=None, sample_weight=None) Incremental fit on a batch of samples

scikitlearn user guide Release 0213

This method is expected to be called several times consecutively on different chunks of a dataset so as to implement outofcore or online learning

This is especially useful when the whole dataset is too big to fit in memory at once

This method has some performance overhead hence it is better to call partialfit on chunks of data that are as large as possible as long as fitting in the memory budget to hide the overhead

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target values

classes arraylike shape nclasses defaultNone List of all the classes that can possibly appear in the y vector

Must be provided at the first call to partialfit can be omitted in subsequent calls

sampleweight arraylike shape nsamples defaultNone Weights applied to individual samples 1 for unweighted

Returns

self object

predictselfX

Perform classification on an array of test vectors X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carray shape nsamples Predicted target values for X

predictlogproba selfX

Return logprobability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carraylike shape nsamples nclasses Returns the logprobability of the samples for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

predictproba selfX

Return probability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carraylike shape nsamples nclasses Returns the probability of the samples for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

scoreselfXsampleweightNone

Returns the mean accuracy on the given test data and labels

629sklearnnaivebayes Naive Bayes 2149

scikitlearn user guide Release 0213

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnnaivebayesMultinomialNB

•Outofcore classification of text documents

•Classification of text documents using sparse features

6294sklearnnaivebayes ComplementNB

classsklearnnaivebayes ComplementNB alpha10 fitpriorTrue classpriorNone

normFalse

The Complement Naive Bayes classifier described in Rennie et al 2003

The Complement Naive Bayes classifier was designed to correct the “severe assumptions” made by the standard Multinomial Naive Bayes classifier It is particularly suited for imbalanced data sets

Read more in the User Guide

Parameters

alpha float optional default10 Additive LaplaceLidstone smoothing parameter 0 for no smoothing

fitprior boolean optional defaultTrue Only used in edge case with a single class in the training set

classprior arraylike size nclasses optional defaultNone Prior probabilities of the classes Not used

norm boolean optional defaultFalse Whether or not a second normalization of the weights is performed The default behavior mirrors the implementations found in Mahout and Weka which do not follow the full algorithm described in Table 9 of the paper

Attributes

classlogprior array shape nclasses Smoothed empirical log probability for each class

Only used in edge case with a single class in the training set

2150 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

featurelogprob array shape nclasses nfeatures Empirical weights for class complements

classcount array shape nclasses Number of samples encountered for each class during fitting This value is weighted by the sample weight when provided

featurecount array shape nclasses nfeatures Number of samples encountered for each class feature during fitting This value is weighted by the sample weight when provided

featureall array shape nfeatures Number of samples encountered for each feature during fitting This value is weighted by the sample weight when provided

References

Rennie J D Shih L Teevan J Karger D R 2003 Tackling the poor assumptions of naive bayes text classifiers In ICML V ol 3 pp 616623 <https://people.csail.mit.edu/jrennie/papers/icml03nbpdf>

Examples

```
import numpy as np
X = np.random.randint(5, size=(6, 100))
y = np.array([2, 3, 4, 5, 6])
from sklearn.naive_bayes import ComplementNB
clf = ComplementNB()
clf.fit(X, y)
ComplementNB(alpha=1.0, class_prior=None, fit_prior=True, norm=False)
print(clf.predict(X[2:3]))
```

Methods

fitself X y sampleweight Fit Naive Bayes classifier according to X y

getparams self deep Get parameters for this estimator

partialfit self X y classes sampleweight Incremental fit on a batch of samples

predict self X Perform classification on an array of test vectors X

predictlogproba self X Return logprobability estimates for the test vector X

predictproba self X Return probability estimates for the test vector X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init self alpha=1.0 fit_prior=True class_prior=None norm=False

fitself X y sampleweight=None

Fit Naive Bayes classifier according to X y

Parameters

X arraylike sparse matrix shape (n_samples, n_features) Training vectors where n_samples is the number of samples and n_features is the number of features

y arraylike shape (n_samples,) Target values

629sklearn.naive_bayes Naive Bayes 2151

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples defaultNone Weights applied to individual samples 1 for unweighted

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit selfXyclassesNone sampleweightNone

Incremental fit on a batch of samples

This method is expected to be called several times consecutively on different chunks of a dataset so as to implement outofcore or online learning

This is especially useful when the whole dataset is too big to fit in memory at once

This method has some performance overhead hence it is better to call partialfit on chunks of data that are as large as possible as long as fitting in the memory budget to hide the overhead

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where

nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target values

classes arraylike shape nclasses defaultNone List of all the classes that can possibly appear in the y vector

Must be provided at the first call to partialfit can be omitted in subsequent calls

sampleweight arraylike shape nsamples defaultNone Weights applied to individual samples 1 for unweighted

Returns

self object

predictselfX

Perform classification on an array of test vectors X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carray shape nsamples Predicted target values for X

predictlogproba selfX

Return logprobability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

2152 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Carraylike shape nsamples nclasses Returns the logprobability of the samples for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

predictproba selfX

Return probability estimates for the test vector X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carraylike shape nsamples nclasses Returns the probability of the samples for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnnaivebayesComplementNB

•Classification of text documents using sparse features

630sklearnneighbors Nearest Neighbors

The sklearnneighbors module implements the knearest neighbors algorithm

User guide See the Nearest Neighbors section for further details

neighborsBallTree BallTree for fast generalized Npoint problems

neighborsDistanceMetric DistanceMetric class

neighborsKDTree KDTree for fast generalized Npoint problems

Continued on next page

630sklearnneighbors Nearest Neighbors 2153

scikitlearn user guide Release 0213

Table 6223 - continued from previous page

neighborsKernelDensity bandwidth Kernel Density Estimation

neighborsKNeighborsClassifier Classifier implementing the knearest neighbors vote

neighborsKNeighborsRegressor nneighbors

Regression based on knearest neighbors

neighborsLocalOutlierFactor nneighbors

Unsupervised Outlier Detection using Local Outlier Factor

LOF

neighborsRadiusNeighborsClassifier Classifier implementing a vote among neighbors within a given radius

neighborsRadiusNeighborsRegressor radius

Regression based on neighbors within a fixed radius

neighborsNearestCentroid metric Nearest centroid classifier

neighborsNearestNeighbors nneighbors

Unsupervised learner for implementing neighbor searches

neighborsNeighborhoodComponentsAnalysis Neighborhood Components Analysis

6301sklearnneighbors BallTree

classsklearnneighbors BallTree

BallTree for fast generalized Npoint problems

BallTreeX leafsize40 metric'minkowski' kwargs

Parameters

Xarraylike shape nsamples nfeatures nsamples is the number of points in the data set and nfeatures is the dimension of the parameter space Note if X is a Ccontiguous array of doubles then data will not be copied Otherwise an internal copy will be made

leafsize positive integer default 40 Number of points at which to switch to bruteforce

Changing leafsize will not affect the results of a query but can significantly impact the speed of a query and the memory required to store the constructed tree The amount of memory needed to store the tree scales as approximately nsamples leafsize For a spec

ifiedleafsize a leaf node is guaranteed to satisfy leafsize npoints

2leafsize except in the case that nsamples leafsize

metric string or DistanceMetric object the distance metric to use for the tree De

fault'minkowski' with p2 that is a euclidean metric See the documentation of the

DistanceMetric class for a list of available metrics balltreevalidmetrics gives a list of the

metrics which are valid for BallTree

Additional keywords are passed to the distance metric class

Attributes

data memory view The training data

Examples

Query for knearest neighbors

import numpy as np

rng np.random.RandomState0

X rng.randomsample10 3 10 points in 3 dimensions

tree BallTreeX leafsize2

dist ind treequeryX1 k3

2154 Chapter 6 API (Application Programming Interface) Reference


```
scikitlearn user guide Release 0213
printind indices of 3 closest neighbors
0 3 1
printdist distances to 3 closest neighbors
0 019662693 029473397
Pickle and Unpickle a tree Note that the state of the tree is saved in the pickle operation the tree needs not be
rebuilt upon unpickling
import numpy as np
import pickle
rng np.random.RandomState(0)
X rng.random(10, 3) 10 points in 3 dimensions
tree BallTree(X, leafsize=2)
s pickle.dumps(tree)
treecopy pickle.loads(s)
dist, ind treecopy.query(X, k=3)
printind indices of 3 closest neighbors
0 3 1
printdist distances to 3 closest neighbors
0 019662693 029473397
Query for neighbors within a given radius
import numpy as np
rng np.random.RandomState(0)
X rng.random(10, 3) 10 points in 3 dimensions
tree BallTree(X, leafsize=2)
print(tree.query_radius(X, r=0.3, count_only=True))
3
ind tree.query_radius(X, r=0.3)
printind indices of neighbors within distance 0.3
3 0 1
Compute a gaussian kernel density estimate
import numpy as np
rng np.random.RandomState(42)
X rng.random(100, 3)
tree BallTree(X)
tree.kernel_density(X, h=0.1, kernel=gaussian)
array(694114649 783281226 72071716)
Compute a twopoint autocorrelation function
import numpy as np
rng np.random.RandomState(0)
X rng.random(30, 3)
r np.linspace(0, 1, 5)
tree BallTree(X)
tree.two_point_correlation(X, r)
array(30 62 278 580 820)
Methods
630 sklearn.neighbors Nearest Neighbors 2155
```

scikitlearn user guide Release 0213

kerneldensity self X h kernel atol Compute the kernel density estimate at points X with the given kernel using the distance metric specified at tree creation

query X k returndistance dualtree query the tree for the k nearest neighbors

queryradius queryradiusself X r countonly False

twopointcorrelation Compute the twopoint correlation function

getarrays

getncalls

gettreestats

resetncalls

init selfargs kwargs

Initialize self See helptypeself for accurate signature

kerneldensity selfXhkernel'gaussian' atol0 rtol1E8 breadthfirstTrue re

turnlogFalse

Compute the kernel density estimate at points X with the given kernel using the distance metric specified at tree creation

Parameters

Xarraylike shape nsamples nfeatures An array of points to query Last dimension

should match dimension of training data

hfloat the bandwidth of the kernel

kernel string specify the kernel to use Options are 'gaussian' 'tophat' 'epanechnikov'

'exponential' 'linear' 'cosine' Default is kernel 'gaussian'

atol rtol float default 0 Specify the desired relative and absolute tolerance of the re

sult If the true result is Ktrue then the returned result Kret satisfies absKtrue

Kret atol rtol Kret The default is zero ie machine precision for

both

breadthfirst boolean default False if True use a breadthfirst search If False default

use a depthfirst search Breadthfirst is generally faster for compact kernels andor high

tolerances

returnlog boolean default False return the logarithm of the result This can be more

accurate than returning the result itself for narrow kernels

Returns

density ndarray The array of logdensity evaluations shape Xshape1

queryXk1returndistanceTrue dualtreeFalse breadthfirstFalse

query the tree for the k nearest neighbors

Parameters

Xarraylike shape nsamples nfeatures An array of points to query

kinteger default 1 The number of nearest neighbors to return

returndistance boolean default True if True return a tuple d i of distances and

indices if False return array i

2156 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

dualtree boolean default False if True use the dual tree formalism for the query a tree is built for the query points and the pair of trees is used to efficiently search this space This can lead to better performance as the number of points grows large
breadthfirst boolean default False if True then query the nodes in a breadthfirst manner Otherwise query the nodes in a depthfirst manner
sortresults boolean default True if True then distances and indices of each point are sorted on return so that the first column contains the closest points Otherwise neighbors are returned in an arbitrary order

Returns

iif returndistance False
di if returndistance True
darray of doubles shape xshape1 k each entry gives the list of distances to the neighbors of the corresponding point
iarray of integers shape xshape1 k each entry gives the list of indices of neighbors of the corresponding point
queryradius
queryradiusself X r countonly False
query the tree for neighbors within a radius r

Parameters

Xarraylike shape nsamples nfeatures An array of points to query
rdistance within which neighbors are returned r can be a single value or an array of values of shape xshape1 if different radii are desired for each point
returndistance boolean default False if True return distances to neighbors of each point if False return only neighbors Note that unlike the query method setting returndistanceTrue here adds to the computation time Not all distances need to be calculated explicitly for returndistanceFalse Results are not sorted by default see sortresults keyword
countonly boolean default False if True return only the count of points within distance r if False return the indices of all points within distance r If returndistanceTrue setting countonlyTrue will result in an error
sortresults boolean default False if True the distances and indices will be sorted before being returned If False the results will not be sorted If returndistance False setting sortresults True will result in an error

Returns

count if countonly True
ind if countonly False and returndistance False
ind dist if countonly False and returndistance True
count array of integers shape Xshape1 each entry gives the number of neighbors within a distance r of the corresponding point
ind array of objects shape Xshape1 each element is a numpy integer array listing the indices of neighbors of the corresponding point Note that unlike the results of a k neighbors query the returned neighbors are not sorted by distance by default
630sklearnneighbors Nearest Neighbors 2157

scikitlearn user guide Release 0213

dist array of objects shape Xshape1 each element is a numpy double array listing the distances corresponding to indices in i

twopointcorrelation

Compute the twopoint correlation function

Parameters

Xarraylike shape nsamples nfeatures An array of points to query Last dimension should match dimension of training data

rarraylike A onedimensional array of distances

dualtree boolean default False If true use a dualtree algorithm Otherwise use a singletree algorithm Dual tree algorithms can have better scaling for large N

Returns

counts ndarray counts[i] contains the number of pairs of points with distance less than or equal to ri

6302sklearnneighbors DistanceMetric

classsklearnneighbors DistanceMetric

DistanceMetric class

This class provides a uniform interface to fast distance metric functions The various metrics can be accessed via thegetmetric class method and the metric string identifier see below For example to use the Euclidean distance

dist DistanceMetricgetmetriceuclidean

X 0 1 2

3 4 5

distpairwiseX

array 0 519615242

519615242 0

Available Metrics

The following lists the string metric identifiers and the associated distance metric classes

Metrics intended for realvalued vector spaces

2158 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

identifier class name args distance function

“euclidean” EuclideanDistance

•sqrtsumx

y2

“manhattan” ManhattanDistance

•sumx y

“chebyshev” ChebyshevDistance

•maxx y

“minkowski” MinkowskiDistance p sumx

yp1p

“wminkowski” WMinkowskiDistance p w sumwx

yp1p

“seuclidean” SEuclideanDistance V sqrtsumx

y2 V

“mahalanobis” MahalanobisDistance V or VI sqrtx y

V1 x y

Metrics intended for twodimensional vector spaces Note that the haversine distance metric requires data in

the form of latitude longitude and both inputs and outputs are in units of radians

identifier class name distance function

“haver

sine”HaversineDis

tance2 arcsinsqrtsin205 dx

cosx1cosx2sin205 dy

Metrics intended for integervalued vector spaces Though intended for integervalued vectors these are also

valid metrics in the case of realvalued vectors

identifier class name distance function

“hamming” HammingDistance Nunequalx y Ntot

“canberra” CanberraDistance sumx y x y

“braycurtis” BrayCurtisDistance sumx y sumx sumy

Metrics intended for booleanvalued vector spaces Any nonzero entry is evaluated to “True” In the listings

below the following abbreviations are used

- N number of dimensions
 - NTT number of dims in which both values are True
 - NTF number of dims in which the first value is True second is False
 - NFT number of dims in which the first value is False second is True
 - NFF number of dims in which both values are False
 - NNEQ number of nonequal dimensions NNEQ NTF NFT
 - NNZ number of nonzero dimensions NNZ NTF NFT NTT
- 630sklearnneighbors Nearest Neighbors 2159

scikitlearn user guide Release 0213

identifier class name distance function

"jaccard" JaccardDistance NNEQ NNZ

"matching" MatchingDistance NNEQ N

"dice" DiceDistance NNEQ NTT NNZ

"kulsinski" KulsinskiDistance NNEQ N NTT NNEQ N

"rogerstanimoto" RogersTanimotoDistance 2 NNEQ N NNEQ

"russellrao" RussellRaoDistance NNZ N

"sokalmichener" SokalMichenerDistance 2 NNEQ N NNEQ

"sokalsneath" SokalSneathDistance NNEQ NNEQ 05 NTT

Userdefined distance

identifier class name args

"pyfunc" PyFuncDistance func

Herefunc is a function which takes two onedimensional numpy arrays and returns a distance Note that in order to be used within the BallTree the distance must be a true metric ie it must satisfy the following properties

1 Nonnegativity $dx \geq 0$

2 Identity $dx = 0$ if and only if $x = y$

3 Symmetry $dx_{y \rightarrow dy} = x$

4 Triangle Inequality $dx_{y \rightarrow dy} \leq z \rightarrow dx \leq z$

Because of the Python object overhead involved in calling the python function this will be fairly slow but it will have the same scaling as other distances

Methods

disttordist Convert the true distance to the reduced distance

getmetric Get the given distance metric from the string identifier

pairwise Compute the pairwise distances between X and Y

rdisttodist Convert the Reduced distance to the true distance

init selfargs kwargs

Initialize self See helptypeself for accurate signature

disttordist

Convert the true distance to the reduced distance

The reduced distance defined for some metrics is a computationally more efficient measure which preserves the rank of the true distance For example in the Euclidean distance metric the reduced distance is the squaredeuclidean distance

getmetric

Get the given distance metric from the string identifier

See the docstring of DistanceMetric for a list of available metrics

Parameters

metric string or class name The distance metric to use

2160 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

kwargs additional arguments will be passed to the requested metric

pairwise

Compute the pairwise distances between X and Y

This is a convenience routine for the sake of testing For many metrics the utilities in

scipyspatialdistancecdist and scipyspatialdistancepdist will be faster

Parameters

Xarraylike Array of shape Nx D representing Nx points in D dimensions

Yarraylike optional Array of shape Ny D representing Ny points in D dimensions

If not specified then YX

Returns

dist ndarray The shape Nx Ny array of pairwise distances between points in X and Y

rdisttodist

Convert the Reduced distance to the true distance

The reduced distance defined for some metrics is a computationally more efficient measure which pre

serves the rank of the true distance For example in the Euclidean distance metric the reduced distance is

the squaredeuclidean distance

6303sklearnneighbors KDTree

classsklearnneighbors KDTree

KDTree for fast generalized Npoint problems

KDTreeX leafsize40 metric'minkowski' kwargs

Parameters

Xarraylike shape nsamples nfeatures nsamples is the number of points in the data

set and nfeatures is the dimension of the parameter space Note if X is a Ccontiguous

array of doubles then data will not be copied Otherwise an internal copy will be made

leafsize positive integer default 40 Number of points at which to switch to bruteforce

Changing leafsize will not affect the results of a query but can significantly impact the

speed of a query and the memory required to store the constructed tree The amount of

memory needed to store the tree scales as approximately nsamples leafsize For a spec

ifiedleafsize a leaf node is guaranteed to satisfy leafsize npoints

2leafsize except in the case that nsamples leafsize

metric string or DistanceMetric object the distance metric to use for the tree De

fault'minkowski' with p2 that is a euclidean metric See the documentation of the

DistanceMetric class for a list of available metrics kdtreevalidmetrics gives a list of the

metrics which are valid for KDTree

Additional keywords are passed to the distance metric class

Attributes

data memory view The training data

6303sklearnneighbors Nearest Neighbors 2161

scikitlearn user guide Release 0213

Examples

Query for knearest neighbors

```
import numpy as np
rng = np.random.RandomState(0)
X = rng.random_sample(10, 3)  # 10 points in 3 dimensions
tree = KDTree(X, leafsize=2)
dist, ind = tree.query(X[1], k=3)
print(ind)  # indices of 3 closest neighbors
0 3 1
print(dist)  # distances to 3 closest neighbors
0 0.19662693 0.29473397
```

Pickle and Unpickle a tree Note that the state of the tree is saved in the pickle operation the tree needs not be rebuilt upon unpickling

```
import numpy as np
import pickle
rng = np.random.RandomState(0)
X = rng.random_sample(10, 3)  # 10 points in 3 dimensions
tree = KDTree(X, leafsize=2)
s = pickle.dumps(tree)
tree_copy = pickle.loads(s)
dist, ind = tree_copy.query(X[1], k=3)
print(ind)  # indices of 3 closest neighbors
0 3 1
print(dist)  # distances to 3 closest neighbors
0 0.19662693 0.29473397
```

Query for neighbors within a given radius

```
import numpy as np
rng = np.random.RandomState(0)
X = rng.random_sample(10, 3)  # 10 points in 3 dimensions
tree = KDTree(X, leafsize=2)
print(tree.query_radius(X[1], r=0.3, count_only=True))
3
ind, tree.query_radius(X[1], r=0.3)
print(ind)  # indices of neighbors within distance 0.3
3 0 1
```

Compute a gaussian kernel density estimate

```
import numpy as np
rng = np.random.RandomState(42)
X = rng.random_sample(100, 3)
tree = KDTree(X)
tree.kernel_density(X[3], h=0.1, kernel=gaussian)
array(6.94114649 7.83281226 7.2071716)
```

Compute a twopoint autocorrelation function

```
import numpy as np
rng = np.random.RandomState(0)
X = rng.random_sample(30, 3)
r = np.linspace(0, 1, 5)
tree = KDTree(X)
```


scikitlearn user guide Release 0213

treetwopointcorrelationX r

array 30 62 278 580 820

Methods

kerneldensity self X h kernel atol Compute the kernel density estimate at points X with the given kernel using the distance metric specified at

tree creation

query X k returndistance dualtree query the tree for the k nearest neighbors

queryradius queryradiussself X r countonly False

twopointcorrelation Compute the twopoint correlation function

getarrays

getncalls

gettreestats

resetncalls

init selfargs kwargs

Initialize self See helptypeself for accurate signature

kerneldensity selfXhkernel'gaussian' atol0 rtol1E8 breadthfirstTrue re

turnlogFalse

Compute the kernel density estimate at points X with the given kernel using the distance metric specified

at tree creation

Parameters

Xarraylike shape nsamples nfeatures An array of points to query Last dimension

should match dimension of training data

hfloat the bandwidth of the kernel

kernel string specify the kernel to use Options are 'gaussian' 'tophat' 'epanechnikov' 'exponential' 'linear' 'cosine' Default is kernel 'gaussian'

atol rtol float default 0 Specify the desired relative and absolute tolerance of the result If the true result is Ktrue then the returned result Kret satisfies absKtrue

Kret atol rtol Kret The default is zero ie machine precision for both

breadthfirst boolean default False if True use a breadthfirst search If False default

use a depthfirst search Breadthfirst is generally faster for compact kernels andor high tolerances

returnlog boolean default False return the logarithm of the result This can be more accurate than returning the result itself for narrow kernels

Returns

density ndarray The array of logdensity evaluations shape Xshape1

queryXk1returndistanceTrue dualtreeFalse breadthfirstFalse

query the tree for the k nearest neighbors

Parameters

630sklearnneighbors Nearest Neighbors 2163

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures An array of points to query
kinteger default 1 The number of nearest neighbors to return
returndistance boolean default True if True return a tuple d i of distances and indices if False return array i
dualtree boolean default False if True use the dual tree formalism for the query a tree is built for the query points and the pair of trees is used to efficiently search this space This can lead to better performance as the number of points grows large
breadthfirst boolean default False if True then query the nodes in a breadthfirst manner Otherwise query the nodes in a depthfirst manner
sortresults boolean default True if True then distances and indices of each point are sorted on return so that the first column contains the closest points Otherwise neighbors are returned in an arbitrary order
Returns
iif returndistance False
di if returndistance True
darray of doubles shape xshape1 k each entry gives the list of distances to the neighbors of the corresponding point
iarray of integers shape xshape1 k each entry gives the list of indices of neighbors of the corresponding point
queryradius
queryradiusself X r countonly False
query the tree for neighbors within a radius r
Parameters
Xarraylike shape nsamples nfeatures An array of points to query
rdistance within which neighbors are returned r can be a single value or an array of values of shape xshape1 if different radii are desired for each point
returndistance boolean default False if True return distances to neighbors of each point if False return only neighbors Note that unlike the query method setting returndistanceTrue here adds to the computation time Not all distances need to be calculated explicitly for returndistanceFalse Results are not sorted by default see sortresults keyword
countonly boolean default False if True return only the count of points within distance r if False return the indices of all points within distance r If returndistanceTrue setting countonlyTrue will result in an error
sortresults boolean default False if True the distances and indices will be sorted before being returned If False the results will not be sorted If returndistance False setting sortresults True will result in an error
Returns
count if countonly True
ind if countonly False and returndistance False
ind dist if countonly False and returndistance True

scikitlearn user guide Release 0213

count array of integers shape Xshape1 each entry gives the number of neighbors within a distance r of the corresponding point
ind array of objects shape Xshape1 each element is a numpy integer array listing the indices of neighbors of the corresponding point Note that unlike the results of a k neighbors query the returned neighbors are not sorted by distance by default
dist array of objects shape Xshape1 each element is a numpy double array listing the distances corresponding to indices in i
twopointcorrelation

Compute the twopoint correlation function

Parameters

Xarraylike shape nsamples nfeatures An array of points to query Last dimension should match dimension of training data

rarraylike A onedimensional array of distances

dualtree boolean default False If true use a dualtree algorithm Otherwise use a singletree algorithm Dual tree algorithms can have better scaling for large N

Returns

counts ndarray counts[i] contains the number of pairs of points with distance less than or equal to r[i]

6304sklearnneighbors KernelDensity

classsklearnneighbors KernelDensity bandwidth10 algorithm'auto' kernel'gaussian'

metric'euclidean' atol0 rtol0 breadthfirstTrue

leafsize40 metricparamsNone

Kernel Density Estimation

Read more in the User Guide

Parameters

bandwidth float The bandwidth of the kernel

algorithm string The tree algorithm to use Valid options are 'kdtree''balltree''auto'

Default is 'auto'

kernel string The kernel to use Valid kernels are 'gaus

sian''tophat''epanechnikov''exponential''linear''cosine' Default is 'gaussian'

metric string The distance metric to use Note that not all metrics are valid with all algorithms

Refer to the documentation of BallTree andKDTree for a description of available algo

rithms Note that the normalization of the density output is correct only for the Euclidean

distance metric Default is 'euclidean'

atol float The desired absolute tolerance of the result A larger tolerance will generally lead

to faster execution Default is 0

rtol float The desired relative tolerance of the result A larger tolerance will generally lead to

faster execution Default is 1E8

breadthfirst boolean If true default use a breadthfirst approach to the problem Otherwise

use a depthfirst approach

leafsize int Specify the leaf size of the underlying tree See BallTree orKDTree for

details Default is 40

630sklearnneighbors Nearest Neighbors 2165

scikitlearn user guide Release 0213

metricparams dict Additional parameters to be passed to the tree for use with the metric
For more information see the documentation of BallTree orKDTree

Methods

fitself X y sampleweight Fit the Kernel Density model on the data

getparams self deep Get parameters for this estimator

sample self nsamples randomstate Generate random samples from the model

score self X y Compute the total log probability density under the model

scoresamples self X Evaluate the density model on the data

setparams self params Set the parameters of this estimator

init selfbandwidth10 algorithm'auto' kernel'gaussian' metric'euclidean' atol0
rtol0 breadthfirstTrue leafsize40 metricparamsNone

fitselfXyNone sampleweightNone

Fit the Kernel Density model on the data

Parameters

Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points
Each row corresponds to a single data point

sampleweight arraylike shape nsamples optional List of sample weights attached to the data X

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

sampleselfnsamples1 randomstateNone

Generate random samples from the model

Currently this is implemented only for gaussian and tophat kernels

Parameters

nsamples int optional Number of samples to generate Defaults to 1

randomstate int RandomState instance or None default to None If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Returns

Xarraylike shape nsamples nfeatures List of samples

scoreselfXyNone

Compute the total log probability density under the model

Parameters

2166 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures List of nfeaturesdimensional data points

Each row corresponds to a single data point

Returns

logprob float Total loglikelihood of the data in X This is normalized to be a probability density so the value will be low for highdimensional data

scoresamples selfX

Evaluate the density model on the data

Parameters

Xarraylike shape nsamples nfeatures An array of points to query Last dimension

should match dimension of training data nfeatures

Returns

density ndarray shape nsamples The array of logdensity evaluations These are normalized to be probability densities so values will be low for highdimensional data

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnneighborsKernelDensity

- Kernel Density Estimation
- Kernel Density Estimate of Species Distributions
- Simple 1D Kernel Density Estimation

6305sklearnneighbors KNeighborsClassifier

classssklearnneighbors KNeighborsClassifier nneighbors5 weights'uniform' al

gorithm'auto' leafsize30 p2met

ric'minkowski' metricparamsNone

njobsNone kwargs

Classifier implementing the knearest neighbors vote

Read more in the User Guide

Parameters

nneighbors int optional default 5 Number of neighbors to use by default for

kneighbors queries

weights str or callable optional default 'uniform' weight function used in prediction Possible values

- 'uniform' uniform weights All points in each neighborhood are weighted equally
- 'distance' weight points by the inverse of their distance in this case closer neighbors of a query point will have a greater influence than neighbors which are further away

630sklearnneighbors Nearest Neighbors 2167

scikitlearn user guide Release 0213

- callable a userdefined function which accepts an array of distances and returns an array of the same shape containing the weights
 - algorithm 'auto' 'balltree' 'kdtree' 'brute' optional Algorithm used to compute the nearest neighbors
 - 'balltree' will use BallTree
 - 'kdtree' will use KDTree
 - 'brute' will use a brute force search
 - 'auto' will attempt to decide the most appropriate algorithm based on the values passed
- tofit method

Note fitting on sparse input will override the setting of this parameter using brute force

leafsize int optional default 30 Leaf size passed to BallTree or KDTree This can affect the speed of the construction and query as well as the memory required to store the tree The optimal value depends on the nature of the problem

p integer optional default 2 Power parameter for the Minkowski metric When p = 1 this is equivalent to using manhattan distance l1 and euclidean distance l2 for p = 2 For arbitrary p minkowski distance lp is used

metric string or callable default 'minkowski' the distance metric to use for the tree The default metric is minkowski and with p2 is equivalent to the standard Euclidean metric See the documentation of the DistanceMetric class for a list of available metrics

metricparams dict optional default None Additional keyword arguments for the metric function

njobs int or None optional default None The number of parallel jobs to run for neighbors search None means 1 unless in a joblibparallelbackend context 1 means using all processors See Glossary for more details Doesn't affect fit method

See also

RadiusNeighborsClassifier

KNeighborsRegressor

RadiusNeighborsRegressor

NearestNeighbors

Notes

See Nearest Neighbors in the online documentation for a discussion of the choice of algorithm and leafsize

Warning Regarding the Nearest Neighbors algorithms if it is found that two neighbors neighbor k1 and k have identical distances but different labels the results will depend on the ordering of the training data

https://en.wikipedia.org/wiki/Knearest_neighbor_algorithm

scikitlearn user guide Release 0213

Examples

```
X 0 1 2 3
y 0 0 1 1
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=3)
neigh.fit(X, y)
print(neigh.predict(1))
0
print(neigh.predict_proba(0))
[[0.66666667 0.33333333]]
```

Methods

```
fit(self, X, y) Fit the model using X as training data and y as target values
get_params(self, deep=True) Get parameters for this estimator
kneighbors(self, X, n_neighbors=5) Finds the Kneighbors of a point
kneighbors_graph(self, X, n_neighbors, mode='distance') Computes the weighted graph of kNeighbors for points in X
predict(self, X) Predict the class labels for the provided data
predict_proba(self, X) Return probability estimates for the test data X
score(self, X, y, sample_weight) Returns the mean accuracy on the given test data and labels
set_params(self, **params) Set the parameters of this estimator
init(self, n_neighbors=5, weights='uniform', algorithm='auto', leafsize=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)
fit(self, X, y) Fit the model using X as training data and y as target values
```

Parameters

```
X arraylike sparse matrix BallTree KDTree Training data If array or matrix shape
n_samples n_features or n_samples n_samples if metric='precomputed'
y arraylike sparse matrix Target values of shape n_samples or n_samples
n_neighbors int, default=5
```

get_params(self, deep=True)

Get parameters for this estimator

Parameters

```
deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators
```

Returns

```
params mapping of string to any Parameter names mapped to their values
```

kneighbors(self, X=None, n_neighbors=None, return_distance=True)

Finds the Kneighbors of a point Returns indices of and distances to the neighbors of each point

Parameters

```
X arraylike sparse matrix Target values of shape n_samples or n_samples
n_neighbors int, default=5
```

scikitlearn user guide Release 0213

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precomputed' The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor
nneighbors int Number of neighbors to get default is the value passed to the constructor
returndistance boolean optional Defaults to True If False distances will not be returned

Returns
dist array Array representing the lengths to points only present if returndistanceTrue
ind array Indices of the nearest points in the population matrix

Examples
In the following example we construct a NeighborsClassifier class from an array representing our data set and ask who's the closest point to 111
samples 0 0 0 0 5 0 1 1 5
from sklearnneighbors import NearestNeighbors
neigh NearestNeighborsnneighbors1
neighfitsamples
NearestNeighborsalgorithmauto leafsize30
printneighkneighbors1 1 1
array05 array2

As you can see it returns 05 and 2 which means that the element is at distance 05 and is the third element of samples indexes start at 0 You can also query for multiple points
X 0 1 0 1 0 1
neighkneighborsX returndistance False
array1
2

kneighborsgraph selfXNone nneighborsNone mode'connectivity'
Computes the weighted graph of kNeighbors for points in X

Parameters
Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precomputed' The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor
nneighbors int Number of neighbors for each sample default is value passed to the constructor
mode 'connectivity' 'distance' optional Type of returned matrix 'connectivity' will return the connectivity matrix with ones and zeros in 'distance' the edges are Euclidean distance between points

Returns
Asparse matrix in CSR (Compressed Sparse Row) format shape nsamples nsamplesfit nsamplesfit is the number of samples in the fitted data A_{i j} is assigned the weight of edge that connects i to j

See also
2170 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

NearestNeighborsradiusneighborsgraph

Examples

X 0 3 1

from sklearnneighbors import NearestNeighbors

neigh = NearestNeighborsnneighbors2

neighfitX

NearestNeighborsalgorithmauto leafsize30

A = neighkneighborsgraphX

Atoarray

array1 0 1

0 1 1

1 0 1

predictselfX

Predict the class labels for the provided data

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precom

puted' Test samples

Returns

yarray of shape nsamples or nsamples noutputs Class labels for each data sample

predictproba selfX

Return probability estimates for the test data X

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precom

puted' Test samples

Returns

parray of shape nsamples nclasses or a list of noutputs of such arrays if noutputs

1 The class probabilities of the input samples Classes are ordered by lexicographic

order

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each

sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

630sklearnneighbors Nearest Neighbors 2171

scikitlearn user guide Release 0213

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnneighborsKNeighborsClassifier

- Classifier comparison
- Plot the decision boundaries of a VotingClassifier
- Digits Classification Exercise
- Nearest Neighbors Classification
- Comparing Nearest Neighbors with and without Neighborhood Components Analysis
- Dimensionality Reduction with Neighborhood Components Analysis
- Classification of text documents using sparse features

6306sklearnneighbors KNeighborsRegressor

classsklearnneighbors KNeighborsRegressor nneighbors5 weights'uniform' algo

rithm'auto' leafsize30 p2met

ric'minkowski' metricparamsNone

njobsNone kwargs

Regression based on knearest neighbors

The target is predicted by local interpolation of the targets associated of the nearest neighbors in the training set

Read more in the User Guide

Parameters

nneighbors int optional default 5 Number of neighbors to use by default for

kneighbors queries

weights str or callable weight function used in prediction Possible values

- 'uniform' uniform weights All points in each neighborhood are weighted equally
- 'distance' weight points by the inverse of their distance in this case closer neighbors of a query point will have a greater influence than neighbors which are further away
- callable a userdefined function which accepts an array of distances and returns an array of the same shape containing the weights

Uniform weights are used by default

algorithm 'auto' 'balltree' 'kdtree' 'brute' optional Algorithm used to compute the nearest neighbors

- 'balltree' will use BallTree
- 'kdtree' will use KDTree
- 'brute' will use a bruteforce search
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed

tofit method

2172 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Note fitting on sparse input will override the setting of this parameter using brute force
leafsize int optional default 30 Leaf size passed to BallTree or KDTree This can affect
the speed of the construction and query as well as the memory required to store the tree
The optimal value depends on the nature of the problem
pinteger optional default 2 Power parameter for the Minkowski metric When p 1 this
is equivalent to using manhattandistance l1 and euclideandistance l2 for p 2 For
arbitrary p minkowskidistance lp is used
metric string or callable default 'minkowski' the distance metric to use for the tree The
default metric is minkowski and with p2 is equivalent to the standard Euclidean metric
See the documentation of the DistanceMetric class for a list of available metrics
metricparams dict optional default None Additional keyword arguments for the metric
function
njobs int or None optional defaultNone The number of parallel jobs to run for neighbors
searchNone means 1 unless in a joblibparallelbackend context1means
using all processors See Glossary for more details Doesn't affect fit method

See also

NearestNeighbors
RadiusNeighborsRegressor
KNeighborsClassifier
RadiusNeighborsClassifier

Notes

See Nearest Neighbors in the online documentation for a discussion of the choice of algorithm and

leafsize

Warning Regarding the Nearest Neighbors algorithms if it is found that two neighbors neighbor k1 and
k have identical distances but different labels the results will depend on the ordering of the training data
https://en.wikipedia.org/wiki/Knearestneighbor_algorithm

Examples

```
X = [[0, 1, 2, 3],
      [0, 0, 1, 1]]
y = [0, 0, 1, 1]
from sklearn.neighbors import KNeighborsRegressor
neigh = KNeighborsRegressor(n_neighbors=2)
neigh.fit(X, y)
KNeighborsRegressor
print(neigh.predict([1.5,
0.5]))
630sklearn.neighbors.Nearest Neighbors 2173
```

Methods

fitself X y Fit the model using X as training data and y as target values

getparams self deep Get parameters for this estimator

kneighbors self X nneighbors Finds the Kneighbors of a point

kneighborsgraph self X nneighbors mode Computes the weighted graph of kNeighbors for points in X

predict self X Predict the target for the provided data

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfnneighbors5 weights'uniform' algorithm'auto' leafsize30 p2metric'minkowski' metricparamsNone njobsNone kwargs

fitselfXy

Fit the model using X as training data and y as target values

Parameters

Xarraylike sparse matrix BallTree KDTree Training data If array or matrix shape nsamples nfeatures or nsamples nsamples if metric'precomputed'

yarraylike sparse matrix

Target values array of float values shape nsamples or nsamples noutputs

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

kneighbors selfXNone nneighborsNone returndistanceTrue

Finds the Kneighbors of a point Returns indices of and distances to the neighbors of each point

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precomputed' The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor

nneighbors int Number of neighbors to get default is the value passed to the constructor

returndistance boolean optional Defaults to True If False distances will not be returned

Returns

dist array Array representing the lengths to points only present if returndistanceTrue

ind array Indices of the nearest points in the population matrix

Examples

In the following example we construct a NeighborsClassifier class from an array representing our data set and ask who's the closest point to 111

```
samples = 0 0 0 0 5 0 1 1 5
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(nneighbors=1)
neigh.fit(samples)
NearestNeighbors(algorithm='auto', leafsize=30)
print(neigh.kneighbors(1, 1, 1))
array([0.5, array(2)])
```

As you can see it returns 0.5 and 2 which means that the element is at distance 0.5 and is the third element of samples indexes start at 0. You can also query for multiple points

```
X = 0 1 0 1 0 1
neigh.kneighbors(X, return_distance=False)
array([1, 2])
```

neigh.kneighbors_graph(self, X=None, n_neighbors=None, mode='connectivity')
Computes the weighted graph of kNeighbors for points in X

Parameters
X: array-like shape (n_query, n_features) or (n_query, n_indexed) if metric = 'precomputed'. The query point or points. If not provided neighbors of each indexed point are returned. In this case the query point is not considered its own neighbor.
n_neighbors: int. Number of neighbors for each sample. Default is value passed to the constructor.
mode: 'connectivity' or 'distance'. optional. Type of returned matrix. 'connectivity' will return the connectivity matrix with ones and zeros. In 'distance' the edges are Euclidean distance between points.

Returns
As sparse matrix in CSR (Compressed Sparse Row) format shape (n_samples, n_samples_fit). n_samples_fit is the number of samples in the fitted data. A[i, j] is assigned the weight of edge that connects i to j.

See also
NearestNeighbors.radius_neighbors_graph

```
Examples
X = 0 3 1
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(nneighbors=2)
neigh.fit(X)
NearestNeighbors(algorithm='auto', leafsize=30)
A = neigh.kneighbors_graph(X)
A.toarray()
array([[0, 1, 1],
       [1, 0, 1],
       [1, 1, 0]])
630sklearn.neighbors.NearestNeighbors(2175)
```

scikitlearn user guide Release 0.21.3

0.1.1

1.0.1

predictselfX

Predict the target for the provided data

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precom

puted' Test samples

Returns

yarray of int shape nsamples or nsamples noutputs Target values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{true\text{mean}}$ 2sum The best possible score

is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 0.23 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnneighborsKNeighborsRegressor

•Face completion with a multioutput estimators

2176 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- Imputing missing values with variants of IterativeImputer
- Nearest Neighbors regression

6307sklearnneighbors LocalOutlierFactor

classsklearnneighbors LocalOutlierFactor nneighbors20 algorithm'auto'

leafsize30 metric'minkowski' p2met

ricparamsNone contamination'legacy'

noveltyFalse njobsNone

Unsupervised Outlier Detection using Local Outlier Factor LOF

The anomaly score of each sample is called Local Outlier Factor It measures the local deviation of density of a given sample with respect to its neighbors It is local in that the anomaly score depends on how isolated the object is with respect to the surrounding neighborhood More precisely locality is given by knearest neighbors whose distance is used to estimate the local density By comparing the local density of a sample to the local densities of its neighbors one can identify samples that have a substantially lower density than their neighbors These are considered outliers

Parameters

nneighbors int optional default20 Number of neighbors to use by default for kneighbors queries If nneighbors is larger than the number of samples provided all samples will be used

algorithm 'auto' 'balltree' 'kdtree' 'brute' optional Algorithm used to compute the nearest neighbors

- 'balltree' will use BallTree
- 'kdtree' will use KDTree
- 'brute' will use a brute force search
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed to fit method

Note fitting on sparse input will override the setting of this parameter using brute force

leafsize int optional default30 Leaf size passed to BallTree or KDTree This can affect the speed of the construction and query as well as the memory required to store the tree The optimal value depends on the nature of the problem

metric string or callable default 'minkowski' metric used for the distance computation Any metric from scikitlearn or scipyspatialdistance can be used

If 'precomputed' the training input X is expected to be a distance matrix

If metric is a callable function it is called on each pair of instances rows and the resulting value recorded The callable should take two arrays as input and return one value indicating the distance between them This works for Scipy's metrics but is less efficient than passing the metric name as a string

Valid values for metric are

- from scikitlearn 'cityblock' 'cosine' 'euclidean' 'l1' 'l2' 'manhattan'
- from scipyspatialdistance 'braycurtis' 'canberra' 'chebyshev' 'correlation' 'dice' 'hamming' 'jaccard' 'kulsinski' 'mahalanobis' 'minkowski' 'rogerstanimoto' 'rus sellrao' 'seuclidean' 'sokalmichener' 'sokalsneath' 'sqeuclidean' 'yule'

630sklearnneighbors Nearest Neighbors 2177

scikitlearn user guide Release 0213

See the documentation for scipyspatialdistance for details on these metrics <https://docs.scipy.org/doc/scipy/reference/spatialdistance.html>

integer optional default 2 Parameter for the Minkowski metric from sklearn

metricspairwisepairwisedistances When $p = 1$ this is equivalent to

using manhattandistance l_1 and euclidean distance l_2 for $p = 2$ For arbitrary p

minkowskidistance l_p is used

metricparams dict optional default None Additional keyword arguments for the metric function

contamination float in [0, 0.5] optional default 0.1 The amount of contamination of the data set i.e. the proportion of outliers in the data set When fitting this is used to define the threshold on the decision function If “auto” the decision function threshold is determined as in the original paper

Changed in version 0.20 The default value of contamination will change from 0.1 in 0.20 to auto in 0.22

novelty boolean default False By default LocalOutlierFactor is only meant to be used for outlier detection novelty=False Set novelty to True if you want to use LocalOutlierFactor for novelty detection In this case be aware that that you should only use predict decisionfunction and scoresamples on new unseen data and not on the training set

njobs int or None optional default None The number of parallel jobs to run for neighbors search None means 1 unless in a joblibparallelbackend context1

means using all processors See Glossary for more details Affects only kneighbors and kneighborsgraph methods

Attributes

negativeoutlierfactor numpy array shape (n_samples) The opposite LOF of the training samples The higher the more normal Inliers tend to have a LOF score close to 1 negativeoutlierfactor close to 1 while outliers tend to have a larger LOF score

The local outlier factor LOF of a sample captures its supposed ‘degree of abnormality’ It is the average of the ratio of the local reachability density of a sample and those of its knearest neighbors

nneighbors integer The actual number of neighbors used for kneighbors queries offset float Offset used to obtain binary labels from the raw scores Observations having a negativeoutlierfactor smaller than offset are detected as abnormal The offset is set to 15 inliers score around 1 except when a contamination parameter different than “auto” is provided In that case the offset is defined in such a way we obtain the expected number of outliers in training

References

Rca479bb498411

Methods

fitself X y Fit the model using X as training data

getparams self deep Get parameters for this estimator

Continued on next page

2178 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Table 6230 – continued from previous page

kneighbors self X nneighbors Finds the Kneighbors of a point
kneighborsgraph self X nneighbors mode Computes the weighted graph of kNeighbors for
points in X
setparams self params Set the parameters of this estimator
init selfnneighbors20 algorithm'auto' leafsize30 metric'minkowski' p2met
ricparamsNone contamination'legacy' noveltyFalse njobsNone
decisionfunction

Shifted opposite of the Local Outlier Factor of X

Bigger is better ie large values correspond to inliers

The shift offset allows a zero threshold for being an outlier Only available for novelty detection when
novelty is set to True The argument X is supposed to contain new data if X contains a point from
training it considers the later in its own neighborhood Also the samples in X are not considered in the
neighborhood of any point

Parameters

Xarraylike shape nsamples nfeatures The query sample or samples to compute the
Local Outlier Factor wrt the training samples

Returns

shiftedoppositelofscores array shape nsamples The shifted opposite of the Local
Outlier Factor of each input samples The lower the more abnormal Negative scores
represent outliers positive scores represent inliers

fitselfXyNone

Fit the model using X as training data

Parameters

Xarraylike sparse matrix BallTree KDTree Training data If array or matrix shape

nsamples nfeatures or nsamples nsamples if metric'precomputed'

ylgnored not used present for API (Application Programming Interface) consistency by convention

Returns

self object

fitpredict

"Fits the model to the training set X and returns the labels

Label is 1 for an inlier and 1 for an outlier according to the LOF score and the contamination parameter

Parameters

Xarraylike shape nsamples nfeatures defaultNone The query sample or samples

to compute the Local Outlier Factor wrt to the training samples

ylgnored not used present for API (Application Programming Interface) consistency by convention

Returns

isinlier array shape nsamples Returns 1 for anomaliesoutliers and 1 for inliers

getparams selfdeepTrue

Get parameters for this estimator

Parameters

630sklearnneighbors Nearest Neighbors 2179

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

kneighbors selfXNone nneighborsNone returndistanceTrue

Finds the Kneighbors of a point Returns indices of and distances to the neighbors of each point

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precom

puted' The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor

nneighbors int Number of neighbors to get default is the value passed to the construc

tor returndistance boolean optional Defaults to True If False distances will not be re

turned

Returns

dist array Array representing the lengths to points only present if returndistanceTrue

ind array Indices of the nearest points in the population matrix

Examples

In the following example we construct a NeighborsClassifier class from an array representing our data set

and ask who's the closest point to 111

samples 0 0 0 0 5 0 1 1 5

from sklearnneighbors import NearestNeighbors

neigh NearestNeighborsnneighbors1

neighfitsamples

NearestNeighborsalgorithmauto leafsize30

printneighkneighbors1 1 1

array05 array2

As you can see it returns 05 and 2 which means that the element is at distance 05 and is the third

element of samples indexes start at 0 You can also query for multiple points

X 0 1 0 1 0 1

neighkneighborsX returndistance False

array1

2

kneighborsgraph selfXNone nneighborsNone mode'connectivity'

Computes the weighted graph of kNeighbors for points in X

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precom

puted' The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor

nneighbors int Number of neighbors for each sample default is value passed to the constructor

2180 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

mode 'connectivity' 'distance' optional Type of returned matrix 'connectivity' will return the connectivity matrix with ones and zeros in 'distance' the edges are Euclidean distance between points

Returns

Asparse matrix in CSR (Compressed Sparse Row) format shape (nsamples, nsamples) fit nsamples fit is the number of samples in the fitted data A_{ij} is assigned the weight of edge that connects i to j

See also

NearestNeighbors radius_neighbors graph

Examples

```
X = [[0, 3, 1],
      [1, 0, 1],
      [1, 0, 1]]
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(n_neighbors=2)
neigh.fit(X)
A = neigh.kneighbors_graph(X)
A.toarray()
array([[0, 1, 1],
       [1, 0, 1],
       [1, 0, 1]])
```

predict

Predict the labels 1 inlier 1 outlier of X according to LOF

This method allows to generalize prediction to new observations not in the training set Only available for novelty detection when novelty is set to True

Parameters

X array-like shape (nsamples, nfeatures) The query sample or samples to compute the Local Outlier Factor wrt to the training samples

Returns

is_inlier array shape (nsamples) Returns 1 for anomalies outliers and 1 for inliers

scores array shape (nsamples)

Opposite of the Local Outlier Factor of X

It is the opposite as bigger is better ie large values correspond to inliers

Only available for novelty detection when novelty is set to True The argument X is supposed to contain new data if X contains a point from training it considers the later in its own neighborhood Also the samples in X are not considered in the neighborhood of any point The scores array on training data is available by considering the the negative_outlier_factor_ attribute

Parameters

X array-like shape (nsamples, nfeatures) The query sample or samples to compute the Local Outlier Factor wrt the training samples

Returns

opposite_lof_scores array shape (nsamples) The opposite of the Local Outlier Factor of each input samples The lower the more abnormal

630 sklearn.neighbors.NearestNeighbors 2181

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnneighborsLocalOutlierFactor

•Comparing anomaly detection algorithms for outlier detection on toy datasets

•Outlier detection with Local Outlier Factor LOF

•Novelty detection with Local Outlier Factor LOF

6308sklearnneighbors RadiusNeighborsClassifier
classsklearnneighbors RadiusNeighborsClassifier radius10 weights'uniform' algo

rithm'auto' leafsize30 p2met

ric'minkowski' outlierlabelNone

metricparamsNone njobsNone

kwargs

Classifier implementing a vote among neighbors within a given radius

Read more in the User Guide

Parameters

radius float optional default 10 Range of parameter space to use by default for

radiusneighbors queries

weights str or callable weight function used in prediction Possible values

• 'uniform' uniform weights All points in each neighborhood are weighted equally

• 'distance' weight points by the inverse of their distance in this case closer neighbors

of a query point will have a greater influence than neighbors which are further away

• callable a userdefined function which accepts an array of distances and returns an

array of the same shape containing the weights

Uniform weights are used by default

algorithm 'auto' 'balltree' 'kdtree' 'brute' optional Algorithm used to compute the

nearest neighbors

• 'balltree' will use BallTree

• 'kdtree' will use KDTree

• 'brute' will use a brute force search

• 'auto' will attempt to decide the most appropriate algorithm based on the values passed

tofit method

Note fitting on sparse input will override the setting of this parameter using brute force

2182 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0.21.3

leafsize int optional default 30 Leaf size passed to BallTree or KDTree This can affect the speed of the construction and query as well as the memory required to store the tree The optimal value depends on the nature of the problem

pinteger optional default 2 Power parameter for the Minkowski metric When p = 1 this is equivalent to using manhattandistance l1 and euclidean distance l2 for p = 2 For

arbitrary p minkowskidistance lp is used

metric string or callable default 'minkowski' the distance metric to use for the tree The default metric is minkowski and with p2 is equivalent to the standard Euclidean metric

See the documentation of the DistanceMetric class for a list of available metrics

outlierlabel int optional default None Label which is given for outlier samples samples with no neighbors on given radius If set to None ValueError is raised when outlier is detected

metricparams dict optional default None Additional keyword arguments for the metric function

njobs int or None optional default None The number of parallel jobs to run for neighbors

searchNone means 1 unless in a joblibparallelbackend context 1 means

using all processors See Glossary for more details

See also

KNeighborsClassifier

RadiusNeighborsRegressor

KNeighborsRegressor

NearestNeighbors

Notes

See Nearest Neighbors in the online documentation for a discussion of the choice of algorithm and

leafsize

https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

Examples

X = 0 1 2 3

y = 0 0 1 1

from sklearn.neighbors import RadiusNeighborsClassifier

neigh = RadiusNeighborsClassifier(radius=10)

neigh.fit(X, y)

RadiusNeighborsClassifier

print(neigh.predict(1.5))

0

Methods

630sklearn.neighbors.NearestNeighbors 2183

scikitlearn user guide Release 0213

fitself X y Fit the model using X as training data and y as target values

getparams self deep Get parameters for this estimator

predict self X Predict the class labels for the provided data

radiusneighbors self X radius Finds the neighbors within a given radius of a point or points

radiusneighborsgraph self X radius

modeComputes the weighted graph of Neighbors for points in X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init selfradius10 weights'uniform' algorithm'auto' leafsize30 p2met

ric'minkowski' outlierlabelNone metricparamsNone njobsNone kwargs

fitselfXy

Fit the model using X as training data and y as target values

Parameters

Xarraylike sparse matrix BallTree KDTree Training data If array or matrix shape

nsamples nfeatures or nsamples nsamples if metric'precomputed'

yarraylike sparse matrix Target values of shape nsamples or nsamples

noutputs

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict the class labels for the provided data

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precom

puted' Test samples

Returns

yarray of shape nsamples or nsamples noutputs Class labels for each data sample

radiusneighbors selfXNone radiusNone returndistanceTrue

Finds the neighbors within a given radius of a point or points

Return the indices and distances of each point from the dataset lying in a ball with size radius around

the points of the query array Points lying on the boundary are included in the results

The result points are notnecessarily sorted by distance to their query point

Parameters

2184 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike nsamples nfeatures optional The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor

radius float Limiting distance of neighbors to return default is the value passed to the constructor

returndistance boolean optional Defaults to True If False distances will not be re turned

Returns

dist array shape nsamples of arrays Array representing the distances to each point only present if returndistanceTrue The distance values are computed according to the metric constructor parameter

ind array shape nsamples of arrays An array of arrays of indices of the approximate nearest points from the population matrix that lie within a ball of size radius around the query points

Notes

Because the number of neighbors of each point is not necessarily equal the results for multiple query points cannot be fit in a standard data array For efficiency radiusneighbors returns arrays of objects where each object is a 1D array of indices or distances

Examples

In the following example we construct a NeighborsClassifier class from an array representing our data set and ask who's the closest point to 1 1 1

```
import numpy as np
samples = 0 0 0 0 5 0 1 1 5
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(radius=16)
neigh.fit(samples)
NearestNeighbors(radius=16, algorithm='auto', leafsize=30)
rng = np.random.RandomState(1)
print(np.asarray(rng, dtype='float64').shape)
print(np.asarray(rng, dtype='float64').shape)
1 2
```

The first array returned contains the distances to all points which are closer than 16 while the second array returned contains their indices In general multiple points can be queried at the same time

radiusneighborsgraph selfXNone radiusNone mode'connectivity'

Computes the weighted graph of Neighbors for points in X

Neighborhoods are restricted the points at a distance lower than radius

Parameters

Xarraylike shape nsamples nfeatures optional The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor

radius float Radius of neighborhoods default is the value passed to the constructor

630sklearn.neighbors Nearest Neighbors 2185

scikitlearn user guide Release 0213

mode 'connectivity' 'distance' optional Type of returned matrix 'connectivity' will return the connectivity matrix with ones and zeros in 'distance' the edges are Euclidean distance between points

Returns
Asparse matrix in CSR (Compressed Sparse Row) format shape nsamples nsamples A_{ij} is assigned the weight of edge that connects i to j

See also
kneighborsgraph

Examples

```
X = [[0, 3, 1],
      [1, 0, 0],
      [0, 1, 0]]
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(radius=15)
neigh.fit(X)
NearestNeighbors(algorithm='auto', leafsize=30)
A = neigh.radius_neighbors_graph(X)
```

array([[0, 1],
 [1, 0],
 [0, 1]])
score = self.Xysampleweight=None
Returns the mean accuracy on the given test data and labels
In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters
Xarraylike shape nsamples nfeatures Test samples
yarraylike shape nsamples or nsamples noutputs True labels for X
sampleweight arraylike shape nsamples optional Sample weights

Returns
score float Mean accuracy of selfpredictX wrt y
setparams selfparams
Set the parameters of this estimator
The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns
self
2186 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

6309sklearnneighbors RadiusNeighborsRegressor

classsklearnneighbors RadiusNeighborsRegressor radius10 weights'uniform' algo

rithm'auto' leafsize30 p2met

ric'minkowski' metricparamsNone

njobsNone kwargs

Regression based on neighbors within a fixed radius

The target is predicted by local interpolation of the targets associated of the nearest neighbors in the training set

Read more in the User Guide

Parameters

radius float optional default 10 Range of parameter space to use by default for

radiusneighbors queries

weights str or callable weight function used in prediction Possible values

- 'uniform' uniform weights All points in each neighborhood are weighted equally
- 'distance' weight points by the inverse of their distance in this case closer neighbors of a query point will have a greater influence than neighbors which are further away
- callable a userdefined function which accepts an array of distances and returns an array of the same shape containing the weights

Uniform weights are used by default

algorithm 'auto' 'balltree' 'kdtree' 'brute' optional Algorithm used to compute the nearest neighbors

- 'balltree' will use BallTree
- 'kdtree' will use KDTree
- 'brute' will use a brute force search
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed

tofit method

Note fitting on sparse input will override the setting of this parameter using brute force

leafsize int optional default 30 Leaf size passed to BallTree or KDTree This can affect the speed of the construction and query as well as the memory required to store the tree

The optimal value depends on the nature of the problem

p integer optional default 2 Power parameter for the Minkowski metric When p 1 this is equivalent to using manhattandistance l1 and euclidean distance l2 for p 2 For

arbitrary p minkowskidistance lp is used

metric string or callable default 'minkowski' the distance metric to use for the tree The

default metric is minkowski and with p2 is equivalent to the standard Euclidean metric

See the documentation of the DistanceMetric class for a list of available metrics

metricparams dict optional default None Additional keyword arguments for the metric function

njobs int or None optional defaultNone

The number of parallel jobs to run for neighbors search None means 1 unless in a

joblibparallelbackend context

1means using all processors See Glossary for more details

See also

630sklearnneighbors Nearest Neighbors 2187

scikitlearn user guide Release 0213

NearestNeighbors

KNeighborsRegressor

KNeighborsClassifier

RadiusNeighborsClassifier

Notes

See Nearest Neighbors in the online documentation for a discussion of the choice of algorithm and leafsize

https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

Examples

```
X = [[0, 1, 2, 3],
     [0, 0, 1, 1]]
y = [0, 0, 1, 1]

from sklearn.neighbors import RadiusNeighborsRegressor
neigh = RadiusNeighborsRegressor(radius=10)
neigh.fit(X, y)
RadiusNeighborsRegressor
print(neigh.predict([1.5, 0.5]))
```

Methods

fit(self, X, y) Fit the model using X as training data and y as target values

get_params(self, deep=True) Get parameters for this estimator

predict(self, X) Predict the target for the provided data

radius_neighbors(self, X, radius) Finds the neighbors within a given radius of a point or points

radius_neighbors_graph(self, X, radius) mode Computes the weighted graph of Neighbors for points in X

score(self, X, y, sample_weight) Returns the coefficient of determination R² of the prediction

set_params(self, **params) Set the parameters of this estimator

init(self, radius=10, weights='uniform', algorithm='auto', leafsize=30, p2metric='minkowski', metric_params=None, n_jobs=None, **kwargs)

fit(self, X, y) Fit the model using X as training data and y as target values

Parameters

X array-like sparse matrix, BallTree, KDTree Training data. If array or matrix shape (n_samples, n_features) or (n_samples, n_samples) if metric='precomputed'

y array-like sparse matrix Target values array of float values shape (n_samples) or (n_samples, n_outputs)

2188 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict the target for the provided data

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precomputed' Test samples

Returns

yarray of float shape nsamples or nsamples noutputs Target values

radiusneighbors selfXNone radiusNone returndistanceTrue

Finds the neighbors within a given radius of a point or points

Return the indices and distances of each point from the dataset lying in a ball with size radius around the points of the query array Points lying on the boundary are included in the results

The result points are notnecessarily sorted by distance to their query point

Parameters

Xarraylike nsamples nfeatures optional The query point or points If not provided

neighbors of each indexed point are returned In this case the query point is not considered its own neighbor

radius float Limiting distance of neighbors to return default is the value passed to the constructor

returndistance boolean optional Defaults to True If False distances will not be returned

Returns

dist array shape nsamples of arrays Array representing the distances to each point

only present if returndistanceTrue The distance values are computed according to the metric constructor parameter

ind array shape nsamples of arrays An array of arrays of indices of the approximate nearest points from the population matrix that lie within a ball of size radius around the query points

Notes

Because the number of neighbors of each point is not necessarily equal the results for multiple query points cannot be fit in a standard data array For efficiency radiusneighbors returns arrays of objects where each object is a 1D array of indices or distances

630sklearnneighbors Nearest Neighbors 2189

Examples

In the following example we construct a NeighborsClassifier class from an array representing our data set and ask who's the closest point to 1 1 1

```
import numpy as np
samples = 0 0 0 0 5 0 1 1 5
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(radius=16)
neigh.fit(samples)
NearestNeighbors(algorithm='auto', leafsize=30)
rng = neigh.radius_neighbors(1, 1, 1)
print(np.asarray(rng, dtype=object))
[[15, 05],
 [1, 2]]
```

The first array returned contains the distances to all points which are closer than 16 while the second array returned contains their indices. In general, multiple points can be queried at the same time.

radius_neighbors_graph(self, X=None, radius=None, mode='connectivity')

Computes the weighted graph of Neighbors for points in X.

Neighborhoods are restricted to the points at a distance lower than radius.

Parameters

X: array-like shape (n_samples, n_features) optional. The query point or points. If not provided, neighbors of each indexed point are returned. In this case, the query point is not considered its own neighbor.

radius: float. Radius of neighborhoods. Default is the value passed to the constructor.

mode: 'connectivity' or 'distance' optional. Type of returned matrix. 'connectivity' will

return the connectivity matrix with ones and zeros. In 'distance' the edges are Euclidean

distance between points.

Returns

A sparse matrix in CSR (Compressed Sparse Row) format shape (n_samples, n_samples). A_{i,j} is assigned the weight of edge that connects i to j.

See also

kneighbors_graph

Examples

```
X = 0 3 1
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(radius=15)
neigh.fit(X)
NearestNeighbors(algorithm='auto', leafsize=30)
A = neigh.radius_neighbors_graph(X)
A.toarray()
array([[1, 0, 1],
       [0, 1, 0],
       [1, 0, 1]])
```

scikitlearn user guide Release 0213

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

63010sklearnneighbors NearestCentroid

classsklearnneighbors NearestCentroid metric'euclidean' shrinkthresholdNone

Nearest centroid classifier

Each class is represented by its centroid with test samples classified to the class with the nearest centroid

Read more in the User Guide

Parameters

metric string or callable The metric to use when calculating distance between instances in

a feature array If metric is a string or callable it must be one of the options allowed by

metricspairwisepairwisedistances for its metric parameter The centroids for the samples

corresponding to each class is the point from which the sum of the distances according to

the metric of all samples that belong to that particular class are minimized If the "manhat

tan" metric is provided this centroid is the median and for all other metrics the centroid is

now set to be the mean

630sklearnneighbors Nearest Neighbors 2191

scikitlearn user guide Release 0213

shrinkthreshold float optional default None Threshold for shrinking centroids to re
move features

Attributes

centroids arraylike shape nclasses nfeatures Centroid of each class

See also

sklearnneighborsKNeighborsClassifier nearest neighbors classifier

Notes

When used for text classification with tfidf vectors this classifier is also known as the Rocchio classifier

References

Tibshirani R Hastie T Narasimhan B Chu G 2002 Diagnosis of multiple cancer types by shrunken
centroids of gene expression Proceedings of the National Academy of Sciences of the United States of America
9910 65676572 The National Academy of Sciences

Examples

```
from sklearnneighborsnearestcentroid import NearestCentroid
import numpy as np
X nparray1 1 2 1 3 2 1 1 2 1 3 2
y nparray1 1 1 2 2 2
clf NearestCentroid
clffitX y
NearestCentroidmetric'euclidean' shrinkthresholdNone
printclfpredict08 1
1
```

Methods

fitself X y Fit the NearestCentroid model according to the given
training data
getparams self deep Get parameters for this estimator
predict self X Perform classification on an array of test vectors X
score self X y sampleweight Returns the mean accuracy on the given test data and
labels
setparams self params Set the parameters of this estimator
init selfmetric'euclidean' shrinkthresholdNone
fitselfXy
Fit the NearestCentroid model according to the given training data
Parameters
Xarraylike sparse matrix shape nsamples nfeatures Training vector where
nsamples is the number of samples and nfeatures is the number of features Note that
2192 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

centroid shrinking cannot be used with sparse matrices

yarray shape nsamples Target values integers

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Perform classification on an array of test vectors X

The predicted class C for each sample in X is returned

Parameters

Xarraylike shape nsamples nfeatures

Returns

Carray shape nsamples

Notes

If the metric constructor parameter is “precomputed” X is assumed to be the distance matrix between the

data to be predicted and selfcentroids

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each

sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it’s possible to update each component

of a nested object

Returns

self

630sklearnneighbors Nearest Neighbors 2193

scikitlearn user guide Release 0213

Examples using sklearnneighborsNearestCentroid

- Nearest Centroid Classification
- Classification of text documents using sparse features

63011sklearnneighbors NearestNeighbors

classsklearnneighbors NearestNeighbors nneighbors5 radius10 algorithm'auto'

leafsize30 metric'minkowski' p2met

ricparamsNone njobsNone kwargs

Unsupervised learner for implementing neighbor searches

Read more in the User Guide

Parameters

nneighbors int optional default 5 Number of neighbors to use by default for
kneighbors queries

radius float optional default 10 Range of parameter space to use by default for

radiusneighbors queries

algorithm 'auto' 'balltree' 'kdtree' 'brute' optional Algorithm used to compute the
nearest neighbors

- 'balltree' will use BallTree
- 'kdtree' will use KDTree
- 'brute' will use a brute force search
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed

tofit method

Note fitting on sparse input will override the setting of this parameter using brute force

leafsize int optional default 30 Leaf size passed to BallTree or KDTree This can affect
the speed of the construction and query as well as the memory required to store the tree

The optimal value depends on the nature of the problem

metric string or callable default 'minkowski' metric to use for distance computation Any

metric from scikitlearn or scipyspatialdistance can be used

If metric is a callable function it is called on each pair of instances rows and the resulting
value recorded The callable should take two arrays as input and return one value indicating
the distance between them This works for Scipy's metrics but is less efficient than passing
the metric name as a string

Distance matrices are not supported

Valid values for metric are

- from scikitlearn 'cityblock' 'cosine' 'euclidean' 'l1' 'l2' 'manhattan'
- from scipyspatialdistance 'braycurtis' 'canberra' 'chebyshev' 'correlation' 'dice'
- 'hamming' 'jaccard' 'kulsinski' 'mahalanobis' 'minkowski' 'rogerstanimoto' 'rus
- sellrao' 'seuclidean' 'sokalmichener' 'sokalsneath' 'sqeuclidean' 'yule'

See the documentation for scipyspatialdistance for details on these metrics

2194 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0.21.3

`p`: integer optional default 2 Parameter for the Minkowski metric from `sklearn.metrics.pairwise_distances`. When `p = 1` this is equivalent to using `manhattan_distance` `l1` and `euclidean_distance` `l2` for `p = 2`. For arbitrary `p` `minkowski_distance` `lp` is used.

`metric_params`: dict optional default None Additional keyword arguments for the metric function.

`n_jobs`: int or None optional default None The number of parallel jobs to run for neighbors.

`search`: None means 1 unless in a `joblib.parallel_backend` context. 1 means using all processors. See Glossary for more details.

See also

- `KNeighborsClassifier`
- `RadiusNeighborsClassifier`
- `KNeighborsRegressor`
- `RadiusNeighborsRegressor`
- `BallTree`

Notes

See Nearest Neighbors in the online documentation for a discussion of the choice of algorithm and `leafsize`.

https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

Examples

```
import numpy as np
from sklearn.neighbors import NearestNeighbors

samples = [[0, 0, 2, 1, 0, 0, 0, 1],
            [0, 0, 0, 0, 0, 0, 0, 0]]
neigh = NearestNeighbors(2, 0.4)
neigh.fit(samples)

neigh.kneighbors([0, 0, 1, 3, 2], return_distance=False)

array([[2, 0],
       [1, 3]])
nbrs = neigh.radius_neighbors([0, 0, 1, 3, 4], return_distance=False)
np.asarray(nbrs[0])
array([2, 0])
```

Methods

`fit(X, y)` Fit the model using `X` as training data.

Continued on next page

630sklearn.neighbors Nearest Neighbors 2195

getparams self deep Get parameters for this estimator

kneighbors self X nneighbors Finds the Kneighbors of a point

kneighborsgraph self X nneighbors mode Computes the weighted graph of kNeighbors for points in X

radiusneighbors self X radius Finds the neighbors within a given radius of a point or points

radiusneighborsgraph self X radius

modeComputes the weighted graph of Neighbors for points in X

setparams self params Set the parameters of this estimator

init selfnneighbors5 radius10 algorithm'auto' leafsize30 metric'minkowski'

p2metricparamsNone njobsNone kwargs

fitselfXyNone

Fit the model using X as training data

Parameters

Xarraylike sparse matrix BallTree KDTree Training data If array or matrix shape nsamples nfeatures or nsamples nsamples if metric'precomputed'

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

kneighbors selfXNone nneighborsNone returndistanceTrue

Finds the Kneighbors of a point Returns indices of and distances to the neighbors of each point

Parameters

Xarraylike shape nquery nfeatures or nquery nindexed if metric 'precomputed' The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor

nneighbors int Number of neighbors to get default is the value passed to the constructor

returndistance boolean optional Defaults to True If False distances will not be returned

Returns

dist array Array representing the lengths to points only present if returndistanceTrue

ind array Indices of the nearest points in the population matrix

Examples

In the following example we construct a NeighborsClassifier class from an array representing our data set and ask who's the closest point to 111

2196 Chapter 6 API (Application Programming Interface) Reference

```
scikitlearn user guide Release 0213
samples 0 0 0 0 5 0 1 1 5
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(nneighbors=1)
neigh.fit(samples)
NearestNeighbors(algorithm='auto', leafsize=30)
print(neigh.kneighbors(1, 1, 1))
array([0.5, 2])
As you can see it returns 0.5 and 2 which means that the element is at distance 0.5 and is the third
element of samples indexes start at 0 You can also query for multiple points
X = [[0, 1, 0, 1, 0, 1]]
neigh.kneighbors(X, return_distance=False)
array([1, 2])
kneighbors_graph(self, X=None, n_neighbors=None, mode='connectivity')
Computes the weighted graph of kNeighbors for points in X
Parameters
X array-like shape (nquery, nfeatures) or (nquery, nindexed) if metric 'precomputed'
The query point or points If not provided neighbors of each indexed point are
returned In this case the query point is not considered its own neighbor
n_neighbors int Number of neighbors for each sample default is value passed to the
constructor
mode {'connectivity', 'distance'} optional Type of returned matrix 'connectivity' will
return the connectivity matrix with ones and zeros in 'distance' the edges are Euclidean
distance between points
Returns
A sparse matrix in CSR (Compressed Sparse Row) format shape (nsamples, nsamples_fit) nsamples_fit is the
number of samples in the fitted data A[i, j] is assigned the weight of edge that connects i
to j
See also
NearestNeighbors.radius_neighbors_graph
Examples
X = [[0, 3, 1]]
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(nneighbors=2)
neigh.fit(X)
NearestNeighbors(algorithm='auto', leafsize=30)
A = neigh.kneighbors_graph(X)
A.toarray()
array([[0, 1],
       [0, 1, 1],
       [1, 0, 1]])
radius_neighbors(self, X=None, radius=None, return_distance=True)
Finds the neighbors within a given radius of a point or points
630 sklearn.neighbors.NearestNeighbors 2197
```

scikitlearn user guide Release 0213

Return the indices and distances of each point from the dataset lying in a ball with size radius around the points of the query array Points lying on the boundary are included in the results
The result points are not necessarily sorted by distance to their query point

Parameters

Xarraylike nsamples nfeatures optional The query point or points If not provided
neighbors of each indexed point are returned In this case the query point is not considered its own neighbor
radius float Limiting distance of neighbors to return default is the value passed to the constructor
returndistance boolean optional Defaults to True If False distances will not be returned

Returns

dist array shape nsamples of arrays Array representing the distances to each point only present if returndistanceTrue The distance values are computed according to the metric constructor parameter
ind array shape nsamples of arrays An array of arrays of indices of the approximate nearest points from the population matrix that lie within a ball of size radius around the query points

Notes

Because the number of neighbors of each point is not necessarily equal the results for multiple query points cannot be fit in a standard data array For efficiency radiusneighbors returns arrays of objects where each object is a 1D array of indices or distances

Examples

In the following example we construct a NeighborsClassifier class from an array representing our data set and ask who's the closest point to 1 1 1

```
import numpy as np
samples = 0 0 0 0 5 0 1 1 5
from sklearn.neighbors import NearestNeighbors
neigh = NearestNeighbors(radius=16)
neigh.fit(samples)
NearestNeighbors(algorithm='auto', leafsize=30)
rng = neigh.radius_neighbors(1, 1, 1)
print(np.asarray(rng, dtype=object))
[[ 15  0 5]
 [ 1 2]]
```

The first array returned contains the distances to all points which are closer than 16 while the second array returned contains their indices In general multiple points can be queried at the same time
radiusneighborsgraph selfXNone radiusNone mode'connectivity'

Computes the weighted graph of Neighbors for points in X
Neighborhoods are restricted the points at a distance lower than radius
Parameters

scikitlearn user guide Release 0.21.3

Xarraylike shape n_samples n_features optional The query point or points If not provided neighbors of each indexed point are returned In this case the query point is not considered its own neighbor

radius float Radius of neighborhoods default is the value passed to the constructor

mode 'connectivity' 'distance' optional Type of returned matrix 'connectivity' will

return the connectivity matrix with ones and zeros in 'distance' the edges are Euclidean distance between points

Returns

Sparse matrix in CSR (Compressed Sparse Row) format shape (n_samples, n_samples) A_{ij} is assigned the weight of edge that connects i to j

See also

kneighborsgraph

Examples

X = 0.3 1

from sklearn.neighbors import NearestNeighbors

neigh = NearestNeighbors(radius=15)

neigh.fit(X)

NearestNeighbors(algorithm='auto', leafsize=30)

A = neigh.radius_neighbors_graph(X)

A.toarray()

array([[0, 1],

[0, 1],

[1, 0],

set_params(self, **kwargs)

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form component__parameter so that it's possible to update each component of a nested object

Returns

self

63012sklearn.neighbors.NeighborhoodComponentsAnalysis

class sklearn.neighbors.NeighborhoodComponentsAnalysis(n_components=None)

init('auto', warm_start=False)

max_iter=50 tol=1e-05

callback=None verbose=0

random_state=None

Neighborhood Components Analysis

Neighborhood Component Analysis (NCA) is a machine learning algorithm for metric learning It learns a linear transformation in a supervised fashion to improve the classification accuracy of a stochastic nearest neighbors rule in the transformed space

Read more in the User Guide

630sklearn.neighbors.Nearest Neighbors 2199

Parameters

ncomponents int optional defaultNone Preferred dimensionality of the projected space
If None it will be set to nfeatures
init string or numpy array optional default'auto' Initialization of the linear transformation
Possible options are 'auto' 'pca' 'lda' 'identity' 'random' and a numpy array of shape
nfeaturesa nfeaturesb
'auto' Depending on ncomponents the most reasonable initialization will be chosen
If ncomponents nclasses we use 'lda' as it uses labels information If
not but ncomponents minnfeatures nsamples we use 'pca' as it
projects data in meaningful directions those of higher variance Otherwise we just use
'identity'
'pca' ncomponents principal components of the inputs passed to fit will be used to
initialize the transformation See decompositionPCA
'lda' minncomponents nclasses most discriminative components of the
inputs passed to fit will be used to initialize the transformation If
ncomponents nclasses the rest of the components will be zero See
discriminantanalysisLinearDiscriminantAnalysis
'identity' If ncomponents is strictly smaller than the dimensionality of the inputs
passed to fit the identity matrix will be truncated to the first ncomponents rows
'random' The initial transformation will be a random array of shape ncomponents
nfeatures Each value is sampled from the standard normal distribution
numpy array nfeaturesb must match the dimensionality of the inputs passed to fit
and nfeaturesa must be less than or equal to that If ncomponents is not None
nfeaturesa must match it
warmstart bool optional defaultFalse If True and fit has been called before the solu
tion of the previous call to fit is used as the initial linear transformation ncomponents
and init will be ignored
maxiter int optional default50 Maximum number of iterations in the optimization
tol float optional default1e5 Convergence tolerance for the optimization
callback callable optional defaultNone If not None this function is called after every
iteration of the optimizer taking as arguments the current solution flattened transformation
matrix and the number of iterations This might be useful in case one wants to examine or
store the transformation found after each iteration
verbose int optional default0 If 0 no progress messages will be printed If 1 progress
messages will be printed to stdout If 2 progress messages will be printed and the disp
parameter of scipy.optimize.minimize will be set to verbose 2
randomstate int or numpyRandomState or None optional defaultNone A pseudo ran
dom number generator object or a seed for it if int If initpca randomstate
is used to initialize the random transformation If initpca randomstate is
passed as an argument to PCA (Principal Component Analysis) when initializing the transformation
Attributes
components array shape ncomponents nfeatures The linear transformation learned dur
ing fitting
niter int Counts the number of iterations performed by the optimizer
2200 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

References

Rf9b6baee82291 Rf9b6baee82292

Examples

```
from sklearnneighborsnca import NeighborhoodComponentsAnalysis
from sklearnneighbors import KNeighborsClassifier
from sklearndatasets import loadiris
from sklearnmodelselection import traintestsplit
```

```
X y loadirisreturnXy True
Xtrain Xtest ytrain ytest traintestsplitX y
stratifyf y testsize07 randomstate42
nca NeighborhoodComponentsAnalysisrandomstate42
ncafitXtrain ytrain
NeighborhoodComponentsAnalysis
knn KNeighborsClassifiernneighbors3
knnfitXtrain ytrain
KNeighborsClassifier
printknnscoreXtest ytest
0933333
knnfitncatransformXtrain ytrain
KNeighborsClassifier
printknnscorencatransformXtest ytest
0961904
```

Methods

```
fitself X y Fit the model according to the given training data
fittransform self X y Fit to data then transform it
getparams self deep Get parameters for this estimator
setparams self params Set the parameters of this estimator
transform self X Applies the learned transformation to the given data
init selfncomponentsNone init'auto' warmstartFalse maxiter50 tol1e05 call
backNone verbose0 randomstateNone
fitselfXy
```

Fit the model according to the given training data

Parameters

Xarraylike shape nsamples nfeatures The training samples
yarraylike shape nsamples The corresponding training labels

Returns

self object returns a trained NeighborhoodComponentsAnalysis model

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

630sklearnneighbors Nearest Neighbors 2201

scikitlearn user guide Release 0213

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Applies the learned transformation to the given data

Parameters

Xarraylike shape nsamples nfeatures Data samples

Returns

Xembedded array shape nsamples ncomponents The data samples transformed

Raises

NotFittedError If fit has not been called before

Examples using sklearnneighborsNeighborhoodComponentsAnalysis

- Manifold learning on handwritten digits Locally Linear Embedding Isomap
- Comparing Nearest Neighbors with and without Neighborhood Components Analysis
- Dimensionality Reduction with Neighborhood Components Analysis
- Neighborhood Components Analysis Illustration

neighborsneighborsgraph X nneighbors

Computes the weighted graph of kNeighbors for points

in X

neighborsradiusneighborsgraph X radius Computes the weighted graph of Neighbors for points in

X

2202 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

63013sklearnneighbors kneighborsgraph X nneighbors mode'connectivity' metric'minkowski' p2metricparamsNone in cludeselfFalse njobsNone

Computes the weighted graph of kNeighbors for points in X

Read more in the User Guide

Parameters

Xarraylike or BallTree shape nsamples nfeatures Sample data in the form of a numpy array or a precomputed BallTree

nneighbors int Number of neighbors for each sample

mode 'connectivity' 'distance' optional Type of returned matrix 'connectivity' will return the connectivity matrix with ones and zeros and 'distance' will return the distances between neighbors according to the given metric

metric string default 'minkowski' The distance metric used to calculate the kNeighbors for each sample point The DistanceMetric class gives a list of available metrics The default distance is 'euclidean' 'minkowski' metric with the p param equal to 2

pint default 2 Power parameter for the Minkowski metric When p = 1 this is equivalent to using manhattandistance l1 and euclidean distance l2 for p = 2 For arbitrary p minkowskidistance lp is used

metricparams dict optional additional keyword arguments for the metric function

includeself bool default False Whether or not to mark each sample as the first nearest neighbor to itself If None then True is used for mode'connectivity' and False for mode'distance' as this will preserve backwards compatibility

njobs int or None optional default None The number of parallel jobs to run for neighbors

searchNone means 1 unless in a joblibparallelbackend context lmeans using all processors See Glossary for more details

Returns

Asparse matrix in CSR (Compressed Sparse Row) format shape nsamples nsamples Ai j is assigned the weight of edge that connects i to j

See also

radiusneighborsgraph

Examples

```
X = 0 3 1
from sklearn.neighbors import kneighborsgraph
A = kneighborsgraph(X, 2, mode='connectivity', includeself=True)
A.toarray()
array([[0, 1, 1],
       [1, 0, 1]])
```

630sklearnneighbors Nearest Neighbors 2203

scikitlearn user guide Release 0213

Examples using sklearnneighborskneighborsgraph

- Agglomerative clustering with and without structure
- Hierarchical clustering structured vs unstructured ward
- Comparing different clustering algorithms on toy datasets

63014sklearnneighbors radiusneighborsgraph

sklearnneighbors radiusneighborsgraph X radius mode'connectivity' metric'minkowski' p2metricparamsNone

includeselfFalse njobsNone

Computes the weighted graph of Neighbors for points in X

Neighborhoods are restricted the points at a distance lower than radius

Read more in the User Guide

Parameters

Xarraylike or BallTree shape nsamples nfeatures Sample data in the form of a numpy

array or a precomputed BallTree

radius float Radius of neighborhoods

mode 'connectivity' 'distance' optional Type of returned matrix 'connectivity' will return the connectivity matrix with ones and zeros and 'distance' will return the distances between neighbors according to the given metric

metric string default 'minkowski' The distance metric used to calculate the neighbors within

a given radius for each sample point The DistanceMetric class gives a list of available

metrics The default distance is 'euclidean' 'minkowski' metric with the param equal to 2

pint default 2 Power parameter for the Minkowski metric When p 1 this is equivalent

to using manhattandistance l1 and euclidean distance l2 for p 2 For arbitrary p

minkowskidistance lp is used

metricparams dict optional additional keyword arguments for the metric function

includeself bool defaultFalse Whether or not to mark each sample as the first nearest

neighbor to itself If None then True is used for mode'connectivity' and False for

mode'distance' as this will preserve backwards compatibility

njobs int or None optional defaultNone The number of parallel jobs to run for neighbors

searchNone means 1 unless in a joblibparallelbackend context1means

using all processors See Glossary for more details

Returns

Asparse matrix in CSR (Compressed Sparse Row) format shape nsamples nsamples Ai j is assigned the weight

of edge that connects i to j

See also

kneighborsgraph

2204 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

```
X = 0.3 * 1
from sklearn.neighbors import radius_neighbors_graph
A = radius_neighbors_graph(X, 15, mode='connectivity',
                           include_self=True)
```

```
Attoarray
array([1, 0, 1])
[0, 1, 0]
[1, 0, 1]
```

631sklearnneuralnetwork Neural network models

The sklearnneuralnetwork module includes models based on neural networks

User guide See the Neural network models supervised and Neural network models unsupervised sections for further details

neuralnetworkBernoulliRBM ncomponents

Bernoulli Restricted Boltzmann Machine RBM

neuralnetworkMLPClassifier Multilayer Perceptron classifier

neuralnetworkMLPRegressor Multilayer Perceptron regressor

6311sklearnneuralnetwork BernoulliRBM

class sklearnneuralnetwork BernoulliRBM ncomponents=256 learningrate=0.1

batchsize=10 niter=10 verbose=0 random_state=None

domstate=None

Bernoulli Restricted Boltzmann Machine RBM

A Restricted Boltzmann Machine with binary visible units and binary hidden units. Parameters are estimated using Stochastic Maximum Likelihood (SML) also known as Persistent Contrastive Divergence (PCD).

The time complexity of this implementation is $O(d^2n)$ assuming $d \gg n$ features, n components.

Read more in the User Guide

Parameters

ncomponents int optional Number of binary hidden units

learningrate float optional The learning rate for weight updates. It is highly recommended to tune this hyperparameter. Reasonable values are in the $[10^{-3}, 10^{-1}]$ range.

batchsize int optional Number of examples per minibatch

niter int optional Number of iterations (sweeps) over the training dataset to perform during training

verbose int optional The verbosity level. The default zero means silent mode.

randomstate integer or RandomState optional A random number generator instance to define the state of the random permutations generator. If an integer is given it fixes the seed.

Defaults to the global numpy random number generator.

Attributes

631sklearnneuralnetwork Neural network models 2205

scikitlearn user guide Release 0213

intercepthidden arraylike shape ncomponents Biases of the hidden units

interceptvisible arraylike shape nfeatures Biases of the visible units

components arraylike shape ncomponents nfeatures Weight matrix where nfeatures

in the number of visible units and ncomponents is the number of hidden units

References

1 Hinton G E Osindero S and Teh Y A fast learning algorithm for deep belief nets Neural Compu

tation 18 pp 15271554 <https://www.cs.toronto.edu/~hinton/absps/fastnc.pdf>

2 Tieleman T Training Restricted Boltzmann Machines using Approximations to the Likelihood Gradi

ent International Conference on Machine Learning ICML 2008

Examples

```
import numpy as np
from sklearn.neural_network import BernoulliRBM
```

```
X = np.array([0 0 0 1 1 1 0 1 1 1 1 1])
```

```
model = BernoulliRBM(n_components=2)
```

```
model.fit(X)
```

```
BernoulliRBM(batch_size=10, learning_rate=0.1, n_components=2, n_iter=10,
```

```
random_state=None, verbose=0)
```

Methods

```
fit(self, X, y) Fit the model to the data X
```

```
fit_transform(self, X, y) Fit to data then transform it
```

```
get_params(self, deep=True) Get parameters for this estimator
```

```
gibbs(self, v) Perform one Gibbs sampling step
```

```
partial_fit(self, X, y) Fit the model to the data X which should contain a par
```

```
tial segment of the data
```

```
score_samples(self, X) Compute the pseudolikelihood of X
```

```
set_params(self, **kwargs) Set the parameters of this estimator
```

```
transform(self, X) Compute the hidden layer activation probabilities
```

```
Ph1vX
```

```
init(self, n_components=256, learning_rate=0.1, batch_size=10, n_iter=10, verbose=0, ran
```

```
dom_state=None)
```

```
fit(self, X, y=None)
```

```
Fit the model to the data X
```

Parameters

X arraylike sparse matrix shape nsamples nfeatures Training data

Returns

self BernoulliRBM The fitted model

```
fit_transform(self, X, y=None) fit_params
```

```
Fit to data then transform it
```

2206 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X
Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

gibbssselfv

Perform one Gibbs sampling step

Parameters

varraylike shape nsamples nfeatures Values of the visible layer to start from

Returns

vnew arraylike shape nsamples nfeatures Values of the visible layer after one Gibbs

step

partialfit selfXyNone

Fit the model to the data X which should contain a partial segment of the data

Parameters

Xarraylike shape nsamples nfeatures Training data

Returns

self BernoulliRBM The fitted model

scoresamples selfX

Compute the pseudolikelihood of X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Values of the visible layer

Must be allboolean not checked

Returns

pseudolikelihood arraylike shape nsamples Value of the pseudolikelihood proxy

for likelihood

Notes

This method is not deterministic it computes a quantity called the free energy on X then on a randomly corrupted version of X and returns the log of the logistic function of the difference

631sklearnneuralnetwork Neural network models 2207

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Compute the hidden layer activation probabilities $Ph1vX$

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The data to be transformed

Returns

harray shape nsamples ncomponents Latent representations of the data

Examples using sklearnneuralnetworkBernoulliRBM

•Restricted Boltzmann Machine features for digit classification

6312sklearnneuralnetwork MLPClassifier

classsklearnneuralnetwork MLPClassifier hiddenlayersizes100 activa

tion'relu' solver'adam' alpha00001

batchsize'auto' learningrate'constant'

learningrateinit0001 power05

maxiter200 shuffleTrue ran

domstateNone tol00001 ver

boseFalse warmstartFalse momen

tum09 nesterovsmomentumTrue

earlystoppingFalse validationfraction01

beta109 beta20999 epsilon1e08

niternochange10

Multilayer Perceptron classifier

This model optimizes the logloss function using LBFGS or stochastic gradient descent

New in version 018

Parameters

hiddenlayersizes tuple length nlayers 2 default 100 The ith element represents the

number of neurons in the ith hidden layer

activation 'identity' 'logistic' 'tanh' 'relu' default 'relu' Activation function for the hid

den layer

- 'identity' noop activation useful to implement linear bottleneck returns $fx \cdot x$
- 'logistic' the logistic sigmoid function returns $fx \cdot \frac{1}{1 + \exp x}$
- 'tanh' the hyperbolic tan function returns $fx \cdot \tanh x$
- 'relu' the rectified linear unit function returns $fx \cdot \max(0, x)$

solver 'lbfgs' 'sgd' 'adam' default 'adam' The solver for weight optimization

2208 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- ‘lbfgs’ is an optimizer in the family of quasiNewton methods
- ‘sgd’ refers to stochastic gradient descent
- ‘adam’ refers to a stochastic gradientbased optimizer proposed by Kingma Diederik and Jimmy Ba

Note The default solver ‘adam’ works pretty well on relatively large datasets with thousands of training samples or more in terms of both training time and validation score For small datasets however ‘lbfgs’ can converge faster and perform better

alpha float optional default 0.0001 L2 penalty regularization term parameter

batchsize int optional default ‘auto’ Size of minibatches for stochastic optimizers If the solver is ‘lbfgs’ the classifier will not use minibatch When set to “auto”

batchsizemin200 nsamples

learningrate ‘constant’ ‘invscaling’ ‘adaptive’ default ‘constant’ Learning rate schedule for weight updates

- ‘constant’ is a constant learning rate given by ‘learningrateinit’
- ‘invscaling’ gradually decreases the learning rate at each time step ‘t’ using an inverse scaling exponent of ‘power’ effectivelearningrate learningrateinit powt

power

- ‘adaptive’ keeps the learning rate constant to ‘learningrateinit’ as long as training loss keeps decreasing Each time two consecutive epochs fail to decrease training loss by at least tol or fail to increase validation score by at least tol if ‘earlystopping’ is on the current learning rate is divided by 5

Only used when solversgd

learningrateinit double optional default 0.001 The initial learning rate used It controls the stepsize in updating the weights Only used when solver’sgd’ or ‘adam’

power double optional default 0.5 The exponent for inverse scaling learning rate It is used in updating effective learning rate when the learningrate is set to ‘invscaling’ Only used when solver’sgd’

maxiter int optional default 200 Maximum number of iterations The solver iterates until convergence determined by ‘tol’ or this number of iterations For stochastic solvers ‘sgd’ ‘adam’ note that this determines the number of epochs how many times each data point will be used not the number of gradient steps

shuffle bool optional default True Whether to shuffle samples in each iteration Only used when solver’sgd’ or ‘adam’

randomstate int RandomState instance or None optional default None If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

tolfloat optional default 1e-4 Tolerance for the optimization When the loss or score is not improving by at least tol for niternochange consecutive iterations unless

learningrate is set to ‘adaptive’ convergence is considered to be reached and training stops

verbose bool optional default False Whether to print progress messages to stdout

warmstart bool optional default False When set to True reuse the solution of the previous call to fit as initialization otherwise just erase the previous solution See the Glossary

631sklearnneuralnetwork Neural network models 2209

scikitlearn user guide Release 0213

momentum float default 0.9 Momentum for gradient descent update Should be between 0 and 1 Only used when solver='sgd'

nesterovsmomentum boolean default True Whether to use Nesterov's momentum Only used when solver='sgd' and momentum = 0

earlystopping bool default False Whether to use early stopping to terminate training when validation score is not improving If set to true it will automatically set aside 10% of training data as validation and terminate training when validation score is not improving by at least tol for niter_nocchange consecutive epochs The split is stratified except in a multilabel setting Only effective when solver='sgd' or 'adam'

validationfraction float optional default 0.1 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if earlystopping is True

beta1 float optional default 0.9 Exponential decay rate for estimates of first moment vector in adam should be in [0, 1] Only used when solver='adam'

beta2 float optional default 0.999 Exponential decay rate for estimates of second moment vector in adam should be in [0, 1] Only used when solver='adam'

epsilon float optional default 1e-8 Value for numerical stability in adam Only used when solver='adam'

niter_nocchange int optional default 10 Maximum number of epochs to not meet tol improvement Only effective when solver='sgd' or 'adam'

New in version 0.20

Attributes

classes array or list of array of shape n_classes Class labels for each output

loss float The current loss computed with the loss function

coefs list length n_layers - 1 The i-th element in the list represents the weight matrix corresponding to layer i

intercepts list length n_layers - 1 The i-th element in the list represents the bias vector corresponding to layer i - 1

niter int The number of iterations the solver has ran

n_layers int Number of layers

n_outputs int Number of outputs

out_activation string Name of the output activation function

Notes

MLPClassifier trains iteratively since at each time step the partial derivatives of the loss function with respect to the model parameters are computed to update the parameters

It can also have a regularization term added to the loss function that shrinks model parameters to prevent over fitting

This implementation works with data represented as dense numpy arrays or sparse scipy arrays of floating point values

2210 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

References

Hinton Geoffrey E “Connectionist learning procedures” Artificial intelligence 401 1989 185234
Glorot Xavier and Yoshua Bengio “Understanding the difficulty of training deep feedforward neural net works” International Conference on Artificial Intelligence and Statistics 2010
He Kaiming et al “Delving deep into rectifiers Surpassing humanlevel performance on imagenet classi fication” arXiv preprint arXiv150201852 2015
Kingma Diederik and Jimmy Ba “Adam A method for stochastic optimization” arXiv preprint arXiv14126980 2014

Methods

fitself X y Fit the model to data matrix X and targets y
getparams self deep Get parameters for this estimator
predict self X Predict using the multilayer perceptron classifier
predictlogproba self X Return the log of probability estimates
predictproba self X Probability estimates
score self X y sampleweight Returns the mean accuracy on the given test data and labels
setparams self params Set the parameters of this estimator
init self hiddenlayersizes100 activation’reLU’ solver’adam’ al pha00001 batchsize’auto’ learningrate’constant’ learningrateinit0001 power05 maxiter200 shuffleTrue randomstateNone tol00001 ver boseFalse warmstartFalse momentum09 nesterovsmomentumTrue earlystoppingFalse validationfraction01 beta109 beta20999 epsilon1e08 niternochange10
fitselfXy

Fit the model to data matrix X and targets y

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input data
yarraylike shape nsamples or nsamples noutputs The target values class labels in classification real numbers in regression
Returns

self returns a trained MLP model
getparams selfdeepTrue
Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values

partialfit

Update the model with a single iteration over the given data

scikitlearn user guide Release 0213

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

yarraylike shape nsamples The target values

classes array shape nclasses default None Classes across all calls to partialfit Can be obtained via npuniqueyall where yall is the target vector of the entire dataset

This argument is required for the first call to partialfit and can be omitted in the subsequent calls Note that y doesn't need to contain all labels in classes

Returns

self returns a trained MLP model

predictselfX

Predict using the multilayer perceptron classifier

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

Returns

yarraylike shape nsamples or nsamples nclasses The predicted classes

predictlogproba selfX

Return the log of probability estimates

Parameters

Xarraylike shape nsamples nfeatures The input data

Returns

logyprob arraylike shape nsamples nclasses The predicted logprobability of the sample for each class in the model where classes are ordered as they are in selfclasses Equivalent to logpredictprobaX

predictproba selfX

Probability estimates

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

Returns

yprob arraylike shape nsamples nclasses The predicted probability of the sample for each class in the model where classes are ordered as they are in selfclasses

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

2212 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0.21.3

set_params(self, **kwargs)

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines. The latter have parameters of the form `component__parameter` so that it's possible to update each component of a nested object.

Returns

self

Examples using `sklearn.neural_network.MLPClassifier`

- Classifier comparison
- Visualization of MLP weights on MNIST
- Varying regularization in Multilayer Perceptron
- Compare Stochastic learning strategies for `MLPClassifier`

6313 `sklearn.neural_network.MLPRegressor`

`class sklearn.neural_network.MLPRegressor(hidden_layer_sizes=100, activation='relu', solver='adam', alpha=0.0001,`

`batch_size='auto', learning_rate='constant',`

`learning_rate_init=0.001, power_t=0.5,`

`max_iter=200, shuffle=True, random_state=None,`

`tol=0.0001, verbose=False, warm_start=False,`

`momentum=0.9, nesterovs_momentum=True,`

`early_stopping=False, validation_fraction=0.1,`

`beta_1=0.9, beta_2=0.999, epsilon=1e-08,`

`n_iter_no_change=10`

Multilayer Perceptron regressor

This model optimizes the squared loss using LBFGS or stochastic gradient descent

New in version 0.18

Parameters

`hidden_layer_sizes`: tuple length `n_layers - 2`, default 100. The *i*th element represents the number of neurons in the *i*th hidden layer.

`activation`: 'identity', 'logistic', 'tanh', 'relu', default 'relu'. Activation function for the hidden layer.

• 'identity': no-op activation, useful to implement linear bottleneck, returns $f(x) = x$

• 'logistic': the logistic sigmoid function, returns $f(x) = \frac{1}{1 + \exp(-x)}$

• 'tanh': the hyperbolic tan function, returns $f(x) = \tanh(x)$

• 'relu': the rectified linear unit function, returns $f(x) = \max(0, x)$

`solver`: 'lbfgs', 'sgd', 'adam', default 'adam'. The solver for weight optimization.

• 'lbfgs' is an optimizer in the family of quasi-Newton methods.

• 'sgd' refers to stochastic gradient descent.

6313 `sklearn.neural_network` Neural network models 22/13

scikitlearn user guide Release 0213

- ‘adam’ refers to a stochastic gradientbased optimizer proposed by Kingma Diederik and Jimmy Ba

Note The default solver ‘adam’ works pretty well on relatively large datasets with thou sands of training samples or more in terms of both training time and validation score For small datasets however ‘lbfgs’ can converge faster and perform better

alpha float optional default 00001 L2 penalty regularization term parameter
batchsize int optional default ‘auto’ Size of minibatches for stochastic optimizers If the solver is ‘lbfgs’ the classifier will not use minibatch When set to “auto”

batchsizemin200 nsamples

learningrate ‘constant’ ‘invscaling’ ‘adaptive’ default ‘constant’ Learning rate schedule for weight updates

- ‘constant’ is a constant learning rate given by ‘learningrateinit’
- ‘invscaling’ gradually decreases the learning rate learningrate at each time step ‘t’ using an inverse scaling exponent of ‘powert’ effectivelearningrate learn ingrateinit powt powert

• ‘adaptive’ keeps the learning rate constant to ‘learningrateinit’ as long as training loss keeps decreasing Each time two consecutive epochs fail to decrease training loss by at least tol or fail to increase validation score by at least tol if ‘earlystopping’ is on the current learning rate is divided by 5

Only used when solver’sgd’

learningrateinit double optional default 0001 The initial learning rate used It controls

the stepsize in updating the weights Only used when solver’sgd’ or ‘adam’

powert double optional default 05 The exponent for inverse scaling learning rate It is used in updating effective learning rate when the learningrate is set to ‘invscaling’ Only used when solver’sgd’

maxiter int optional default 200 Maximum number of iterations The solver iterates until convergence determined by ‘tol’ or this number of iterations For stochastic solvers ‘sgd’ ‘adam’ note that this determines the number of epochs how many times each data point will be used not the number of gradient steps

shuffle bool optional default True Whether to shuffle samples in each iteration Only used when solver’sgd’ or ‘adam’

randomstate int RandomState instance or None optional default None If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

tolfloat optional default 1e4 Tolerance for the optimization When the loss or score is not improving by at least tol forniternochange consecutive iterations unless learningrate is set to ‘adaptive’ convergence is considered to be reached and training stops

verbose bool optional default False Whether to print progress messages to stdout

warmstart bool optional default False When set to True reuse the solution of the previous call to fit as initialization otherwise just erase the previous solution See the Glossary

momentum float default 09 Momentum for gradient descent update Should be between 0 and 1 Only used when solver’sgd’

2214 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

nesterovsmomentum boolean default True Whether to use Nesterov’s momentum Only used when solver’sgd’ and momentum 0

earlystopping bool default False Whether to use early stopping to terminate training when validation score is not improving If set to true it will automatically set aside 10 of training data as validation and terminate training when validation score is not improving by at least tol forniternochange consecutive epochs Only effective when solver’sgd’ or ‘adam’

validationfraction float optional default 01 The proportion of training data to set aside as validation set for early stopping Must be between 0 and 1 Only used if earlystopping is True

beta1 float optional default 09 Exponential decay rate for estimates of first moment vector in adam should be in 0 1 Only used when solver’adam’

beta2 float optional default 0999 Exponential decay rate for estimates of second moment vector in adam should be in 0 1 Only used when solver’adam’

epsilon float optional default 1e8 Value for numerical stability in adam Only used when solver’adam’

niternochange int optional default 10 Maximum number of epochs to not meet tol improvement Only effective when solver’sgd’ or ‘adam’

New in version 020

Attributes

loss float The current loss computed with the loss function

coefs list length nlayers 1 The ith element in the list represents the weight matrix corresponding to layer i

intercepts list length nlayers 1 The ith element in the list represents the bias vector corresponding to layer i 1

niter int The number of iterations the solver has ran

nlayers int Number of layers

noutputs int Number of outputs

outactivation string Name of the output activation function

Notes

MLPRegressor trains iteratively since at each time step the partial derivatives of the loss function with respect to the model parameters are computed to update the parameters

It can also have a regularization term added to the loss function that shrinks model parameters to prevent over fitting

This implementation works with data represented as dense and sparse numpy arrays of floating point values

References

Hinton Geoffrey E “Connectionist learning procedures” Artificial intelligence 401 1989 185234

Glorot Xavier and Yoshua Bengio “Understanding the difficulty of training deep feedforward neural net works” International Conference on Artificial Intelligence and Statistics 2010

631sklearnneuralnetwork Neural network models 2215

scikitlearn user guide Release 0213

He Kaiming et al “Delving deep into rectifiers Surpassing humanlevel performance on imagenet classification” arXiv preprint arXiv150201852 2015

Kingma Diederik and Jimmy Ba “Adam A method for stochastic optimization” arXiv preprint arXiv14126980 2014

Methods

fitself X y Fit the model to data matrix X and targets y

getparams self deep Get parameters for this estimator

predict self X Predict using the multilayer perceptron model

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init self hiddenlayersizes100 activation’reLU’ solver’adam’ alpha00001 batchsize’auto’ learningrate’constant’ learningrateinit0001 powert05 maxiter200 shuffleTrue randomstateNone tol00001 verboseFalse warmstartFalse momentum09 nesterovsmomentumTrue earlystoppingFalse validationfraction01 beta109 beta20999 epsilon1e08 niternochange10

fitselfXy

Fit the model to data matrix X and targets y

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input data

yarraylike shape nsamples or nsamples noutputs The target values class labels in classification real numbers in regression

Returns

self returns a trained MLP model

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

partialfit

Update the model with a single iteration over the given data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

yarraylike shape nsamples The target values

Returns

self returns a trained MLP model

2216 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

predictselfX

Predict using the multilayer perceptron model

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data

Returns

yarraylike shape nsamples noutputs The predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnneuralnetworkMLPRegressor

•Partial Dependence Plots

631sklearnneuralnetwork Neural network models 2217

scikitlearn user guide Release 0213

632sklearnpipeline Pipeline

The sklearn pipeline module implements utilities to build a composite estimator as a chain of transforms and estimators

pipelineFeatureUnion transformerlist Concatenates results of multiple transformer objects

pipelinePipeline steps memory verbose Pipeline of transforms with a final estimator

6321sklearnpipeline FeatureUnion

class sklearn pipeline FeatureUnion transformerlist njobsNone transformerweightsNone

verboseFalse

Concatenates results of multiple transformer objects

This estimator applies a list of transformer objects in parallel to the input data then concatenates the results

This is useful to combine several feature extraction mechanisms into a single transformer

Parameters of the transformers may be set using its name and the parameter name separated by a " " A

transformer may be replaced entirely by setting the parameter with its name to another transformer or removed

by setting to 'drop' or None

Read more in the User Guide

Parameters

transformerlist list of string transformer tuples List of transformer objects to be applied

to the data The first half of each tuple is the name of the transformer

njobs int or None optional defaultNone Number of jobs to run in parallel None means 1

unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

transformerweights dict optional Multiplicative weights for features per transformer Keys

are transformer names values the weights

verbose boolean optionaldefaultFalse If True the time elapsed while fitting each trans

former will be printed as it is completed

See also

sklearnpipelinemakeunion convenience function for simplified feature union construction

Examples

from sklearnpipeline import FeatureUnion

from sklearndecomposition import PCA (Principal Component Analysis) TruncatedSVD

union FeatureUnion(pca PCA(ncomponents1

svd TruncatedSVD(ncomponents2

X 0 1 3 2 2 5

unionfittransformX

array 15 30 08

15 57 04

2218 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Methods

fitself X y Fit all transformers using X

fittransform self X y Fit all transformers transform the data and concatenate results

getfeaturenames self Get feature names from all transformers

getparams self deep Get parameters for this estimator

setparams self kwargs Set the parameters of this estimator

transform self X Transform X separately by each transformer concatenate results

init selftransformerlist njobsNone transformerweightsNone verboseFalse

fitselfXyNone

Fit all transformers using X

Parameters

Xiterable or arraylike depending on transformers Input data used to fit transformers

yarraylike shape nsamples optional Targets for supervised learning

Returns

self FeatureUnion This estimator

fittransform selfXyNone fitparams

Fit all transformers transform the data and concatenate results

Parameters

Xiterable or arraylike depending on transformers Input data to be transformed

yarraylike shape nsamples optional Targets for supervised learning

Returns

Xt arraylike or sparse matrix shape nsamples sumncomponents hstack of results

of transformers sumncomponents is the sum of ncomponents output dimension over transformers

getfeaturenames self

Get feature names from all transformers

Returns

featurenames list of strings Names of the features produced by transform

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfkwargs

Set the parameters of this estimator

632sklearnpipeline Pipeline 2219

scikitlearn user guide Release 0213

Valid parameter keys can be listed with getparams

Returns

self

transform selfX

Transform X separately by each transformer concatenate results

Parameters

Xiterable or arraylike depending on transformers Input data to be transformed

Returns

Xt arraylike or sparse matrix shape nsamples sumncomponents hstack of results

of transformers sumncomponents is the sum of ncomponents output dimension over transformers

Examples using sklearnpipelineFeatureUnion

•Concatenating multiple feature extraction methods

6322sklearnpipeline Pipeline

classsklearnpipeline Pipeline steps memoryNone verboseFalse

Pipeline of transforms with a final estimator

Sequentially apply a list of transforms and a final estimator Intermediate steps of the pipeline must be ‘trans forms’ that is they must implement fit and transform methods The final estimator only needs to implement fit

The transformers in the pipeline can be cached using memory argument

The purpose of the pipeline is to assemble several steps that can be crossvalidated together while setting differ ent parameters For this it enables setting parameters of the various steps using their names and the parameter name separated by a ‘’ as in the example below A step’s estimator may be replaced entirely by setting the parameter with its name to another estimator or a transformer removed by setting it to ‘passthrough’ or None

Read more in the User Guide

Parameters

steps list List of name transform tuples implementing fittransform that are chained in the order in which they are chained with the last object an estimator

memory None str or object with the joblibMemory interface optional Used to cache the fit ted transformers of the pipeline By default no caching is performed If a string is given it is the path to the caching directory Enabling caching triggers a clone of the transform ers before fitting Therefore the transformer instance given to the pipeline cannot be in spected directly Use the attribute namedsteps orsteps to inspect estimators within the pipeline Caching the transformers is advantageous when fitting is time consuming verbose boolean optional If True the time elapsed while fitting each step will be printed as it is completed

Attributes

namedsteps bunch object a dictionary with attribute access Readonly attribute to access

any step parameter by user given name Keys are step names and values are steps parameters

See also

2220 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
sklearnpipeline makepipeline convenience function for simplified pipeline construction
Examples

```
from sklearn import svm
from sklearn.datasets import samplesgenerator
from sklearn.featureselection import SelectKBest
from sklearn.featureselection import fregression
from sklearn.pipeline import Pipeline
generate some data to play with
X y samplesgeneratormakeclassification
ninformative5 nredundant0 randomstate42
ANOVA SVMC
anovafilter SelectKBestfregression k5
clf svmSVCKernellinear
anovasvm Pipelineanova anovafilter svc clf
You can set the parameters using the names issued
For instance fit using a k of 10 in the SelectKBest
and a parameter C of the svm
anovasvmsetparamsanovak10 svcC1fitX y
```

```
Pipeline memoryNone
stepsanova SelectKBest
svc SVC (Support Vector Classification) verboseFalse
prediction anovasvmpredictX
anovasvmscoreX y
083
getting the selected features chosen by anovafilter
anovasvmanovagetsupport
```

```
arrayFalse False True True False False True True False
True False True True False True False True True
False False
Another way to get selected features chosen by anovafilter
anovasvmnamedstepsanovagetsupport
```

```
arrayFalse False True True False False True True False
True False True True False True False True True
False False
```

```
Indexing can also be used to extract a subpipeline
subpipeline anovasvm1
subpipeline
Pipeline memoryNone stepsanova verboseFalse
coef anovasvm1coef
anovasvmsvc isanovasvm1
```

```
True
coefshape
1 10
subpipelineinversetransformcoefshape
1 20
```

Methods
632sklearnpipeline Pipeline 2221

scikitlearn user guide Release 0213

decisionfunction self X Apply transforms and decisionfunction of the final estimator

fitself X y Fit the model

fitpredict self X y Applies fitpredict of last step in pipeline after transforms

fittransform self X y Fit the model and transform with the final estimator

getparams self deep Get parameters for this estimator

predict self X predictparams Apply transforms to the data and predict with the final estimator

predictlogproba self X Apply transforms and predictlogproba of the final estimator

predictproba self X Apply transforms and predictproba of the final estimator

score self X y sampleweight Apply transforms and score with the final estimator

setparams self kwargs Set the parameters of this estimator

init selfsteps memoryNone verboseFalse

decisionfunction selfX

Apply transforms and decisionfunction of the final estimator

Parameters

Xiterable Data to predict on Must fulfill input requirements of first step of the pipeline

Returns

yscore arraylike shape nsamples nclasses

fitselfXyNone fitparams

Fit the model

Fit all the transforms one after the other and transform the data then fit the transformed data using the final estimator

Parameters

Xiterable Training data Must fulfill input requirements of first step of the pipeline

yiterable defaultNone Training targets Must fulfill label requirements for all steps of the pipeline

fitparams dict of string object Parameters passed to the fit method of each step

where each parameter name is prefixed such that parameter pfor stepshas keysp

Returns

self Pipeline This estimator

fitpredict selfXyNone fitparams

Applies fitpredict of last step in pipeline after transforms

Applies fittransforms of a pipeline to the data followed by the fitpredict method of the final estimator in the pipeline Valid only if the final estimator implements fitpredict

Parameters

Xiterable Training data Must fulfill input requirements of first step of the pipeline

yiterable defaultNone Training targets Must fulfill label requirements for all steps of the pipeline

2222 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

fitparams dict of string object Parameters passed to the fit method of each step
where each parameter name is prefixed such that parameter pfor stepshas keysp

Returns

ypred arraylike

fittransform selfXyNone fitparams

Fit the model and transform with the final estimator

Fits all the transforms one after the other and transforms the data then uses fittransform on transformed
data with the final estimator

Parameters

Xiterable Training data Must fulfill input requirements of first step of the pipeline

yiterable defaultNone Training targets Must fulfill label requirements for all steps of
the pipeline

fitparams dict of string object Parameters passed to the fit method of each step
where each parameter name is prefixed such that parameter pfor stepshas keysp

Returns

Xtarraylike shape nsamples ntransformedfeatures Transformed samples

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform

Apply inverse transformations in reverse order

All estimators in the pipeline must support inversetransform

Parameters

Xtarraylike shape nsamples ntransformedfeatures Data samples where
nsamples is the number of samples and nfeatures is the number of features Must
fulfill input requirements of last step of pipeline's inversetransform method

Returns

Xtarraylike shape nsamples nfeatures

predictselfXpredictparams

Apply transforms to the data and predict with the final estimator

Parameters

Xiterable Data to predict on Must fulfill input requirements of first step of the pipeline

predictparams dict of string object Parameters to the predict called at the end of
all transformations in the pipeline Note that while this may be used to return uncertainties
from some models with returnstd or returncov uncertainties that are generated by the
transformations in the pipeline are not propagated to the final estimator

Returns

632sklearnpipeline Pipeline 2223

scikitlearn user guide Release 0213

`ypred` arraylike

`predictlogproba` selfX

Apply transforms and `predictlogproba` of the final estimator

Parameters

Xiterable Data to predict on Must fulfill input requirements of first step of the pipeline

Returns

`yscore` arraylike shape `nsamples nclasses`

`predictproba` selfX

Apply transforms and `predictproba` of the final estimator

Parameters

Xiterable Data to predict on Must fulfill input requirements of first step of the pipeline

Returns

`yproba` arraylike shape `nsamples nclasses`

`scoreselfXyNone sampleweightNone`

Apply transforms and score with the final estimator

Parameters

Xiterable Data to predict on Must fulfill input requirements of first step of the pipeline

`yiterable defaultNone` Targets used for scoring Must fulfill label requirements for all steps of the pipeline

`sampleweight arraylike defaultNone` If not `None` this argument is passed as `sampleweight` keyword argument to the score method of the final estimator

Returns

`score` float

`setparams selfkwargs`

Set the parameters of this estimator

Valid parameter keys can be listed with `getparams`

Returns

self

`transform`

Apply transforms and transform with the final estimator

This also works where final estimator is `None` all prior transformations are applied

Parameters

Xiterable Data to transform Must fulfill input requirements of first step of the pipeline

Returns

Xtarraylike shape `nsamples ntransformedfeatures`

2224 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples using sklearnpipelinePipeline

- Explicit feature map approximation for RBF (Radial Basis Function) kernels
- Feature agglomeration vs univariate selection
- Concatenating multiple feature extraction methods
- Pipelining chaining a PCA (Principal Component Analysis) and a logistic regression
- Column Transformer with Mixed Types
- Selecting dimensionality reduction with Pipeline and GridSearchCV
- Column Transformer with Heterogeneous Data Sources
- Underfitting vs Overfitting
- Balance model complexity and crossvalidated score
- Sample pipeline for text feature extraction and evaluation
- Comparing Nearest Neighbors with and without Neighborhood Components Analysis
- Restricted Boltzmann Machine features for digit classification
- SVMAnova SVM (Support Vector Machine) with univariate feature selection
- Classification of text documents using sparse features

pipelinemakepipeline steps kwargs Construct a Pipeline from the given estimators

pipelinemakeunion transformers kwargs Construct a FeatureUnion from the given transformers

6323sklearnpipeline makepipeline

sklearnpipeline makepipeline steps kwargs

Construct a Pipeline from the given estimators

This is a shorthand for the Pipeline constructor it does not require and does not permit naming the estimators

Instead their names will be set to the lowercase of their types automatically

Parameters

steps list of estimators

memory None str or object with the joblibMemory interface optional Used to cache the fit

ted transformers of the pipeline By default no caching is performed If a string is given

it is the path to the caching directory Enabling caching triggers a clone of the transform

ers before fitting Therefore the transformer instance given to the pipeline cannot be in

spected directly Use the attribute namedsteps orsteps to inspect estimators within

the pipeline Caching the transformers is advantageous when fitting is time consuming

verbose boolean optional If True the time elapsed while fitting each step will be printed as it

is completed

Returns

pPipeline

See also

sklearnpipelinePipeline Class for creating a pipeline of transforms with a final estimator

6323sklearnpipeline Pipeline 2225

scikitlearn user guide Release 0213

Examples

```
from sklearnnaivebayes import GaussianNB
from sklearnpreprocessing import StandardScaler
makepipelineStandardScaler GaussianNBpriors None
```

Pipeline memoryNone

stepsstandardscaler

StandardScaler copyTrue with meanTrue with stdTrue

GaussianNB

GaussianNBpriorsNone var smoothing1e09

verboseFalse

Examples using sklearn pipeline makepipeline

- Feature transformations with ensembles of trees
- Pipeline Anova SVM (Support Vector Machine)
- Imputing missing values with variants of IterativeImputer
- Imputing missing values before building an estimator
- Polynomial interpolation
- Robust linear estimator fitting
- Dimensionality Reduction with Neighborhood Components Analysis
- Using FunctionTransformer to select columns
- Importance of Feature Scaling
- Feature discretization
- Clustering text documents using kmeans

6324sklearn pipeline makeunion

sklearn pipeline makeunion transformers kwargs

Construct a FeatureUnion from the given transformers

This is a shorthand for the FeatureUnion constructor it does not require and does not permit naming the transformers Instead they will be given names automatically based on their types It also does not allow weighting

Parameters

transformers list of estimators

njobs int or None optional defaultNone Number of jobs to run in parallel None means 1 unless in a joblibparallelbackend context 1 means using all processors See

Glossary for more details

verbose boolean optional defaultFalse If True the time elapsed while fitting each transformer will be printed as it is completed

Returns

fFeatureUnion

scikitlearn user guide Release 0213

See also

sklearnpipelineFeatureUnion Class for concatenating the results of multiple transformer objects

Examples

from sklearndecomposition import PCA (Principal Component Analysis) TruncatedSVD

from sklearnpipeline import makeunion

makeunionPCA TruncatedSVD

FeatureUnionnjobsNone

transformerlistpca

PCAcopyTrue iteratedpowerauto

ncomponentsNone randomstateNone

svdsolverauto tol00 whitenFalse

truncatedsvd

TruncatedSVDalgorithmrandomized

ncomponents2 niter5

randomstateNone tol00

transformerweightsNone verboseFalse

Examples using sklearnpipelinemakeunion

•Imputing missing values before building an estimator

633sklearninspection inspection

The sklearninspection module includes tools for model inspection

inspectionpartialdependence estimator X

Partial dependence of features

inspectionplotpartialdependence

Partial dependence plots

6331sklearninspection partialdependence

sklearninspection partialdependence estimator Xfeatures responsemethod'auto'

percentiles005 095 gridresolution100

method'auto'

Partial dependence of features

Partial dependence of a feature or a set of features corresponds to the average response of an estimator for each possible value of the feature

Read more in the User Guide

Parameters

estimator BaseEstimator A fitted estimator object implementing predict predictproba or

decisionfunction Multioutputmulticlass classifiers are not supported

Xarraylike shape nsamples nfeatures Xis used both to generate a grid of values for the

features and to compute the averaged predictions when method is 'brute'

633sklearninspection inspection 2227

scikitlearn user guide Release 0213

features list or arraylike of int The target features for which the partial dependency should be computed

response method 'auto' 'predictproba' or 'decisionfunction' optional default 'auto'

Specifies whether to use predictproba or decisionfunction as the target response For regressors this parameter is ignored and the response is always the output of predict By default predictproba is tried first and we revert to decisionfunction if it doesn't exist If method is 'recursion' the response is always the output of decisionfunction

percentiles tuple of float optional default (0.05, 0.95) The lower and upper percentile used to create the extreme values for the grid Must be in [0, 1]

grid resolution int optional default 100 The number of equally spaced points on the grid for each target feature

method str optional default 'auto' The method used to calculate the averaged predictions

- 'recursion' is only supported for objects inheriting from BaseGradientBoosting but is more efficient in terms of speed With this method X is only used to build the grid and the partial dependences are computed using the training data This method does not account for the init predictor of the boosting process which may lead to incorrect values see warning below With this method the target response of a classifier is always the decision function not the predicted probabilities

- 'brute' is supported for any estimator but is more computationally intensive

- If 'auto' then 'recursion' will be used for BaseGradientBoosting estimators with

init=None and 'brute' for all other

Returns

averaged_predictions ndarray shape (n_outputs, len(values_0), len(values_1), ...) The predictions for all the points in the grid averaged over all samples in X or over the training data

if method is 'recursion' n_outputs corresponds to the number of classes in a multi-class setting or to the number of tasks for multioutput regression For classical regression and binary classification n_outputs=1 n_values[feature_j] corresponds to the size of values_j

values seq of 1d ndarrays The values with which the grid has been created The generated grid is a cartesian product of the arrays in values len(values)=len(features)

The size of each array values_j is either grid_resolution or the number of unique values in X[j] whichever is smaller

Warning The 'recursion' method only works for gradient boosting estimators and unlike the 'brute' method it does not account for the init predictor of the boosting process In practice this will produce the same values as 'brute' up to a constant offset in the target response provided that init is a constant estimator which is the default However as soon as init is not a constant estimator the partial dependence values are incorrect for 'recursion'

See also

sklearn.inspection.plot_partial_dependence Plot partial dependence

2228 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

X 0 0 2 1 0 0

y 0 1

from sklearnensemble import GradientBoostingClassifier

gb GradientBoostingClassifierrandomstate0fitX y

partialdependencegb features0 XX percentiles0 1

gridresolution2

array452 452 array 0 1

Examples using sklearninspectionpartialdependence

•Partial Dependence Plots

6332sklearninspection plotpartialdependence

sklearninspection plotpartialdependence estimator Xfeatures featurenamesNone

targetNone responsemethod'auto'

ncols3 gridresolution100 per

centiles005 095 method'auto'

njobsNone verbose0 figNone

linekwNone contourkwNone

Partial dependence plots

Thelenfeatures plots are arranged in a grid with ncols columns Twoway partial dependence plots

are plotted as contour plots

Read more in the User Guide

Parameters

estimator BaseEstimator A fitted estimator object implementing predict predictproba or decisionfunction Multioutputmulticlass classifiers are not supported

Xarraylike shape nsamples nfeatures The data to use to build the grid of values on

which the dependence will be evaluated This is usually the training data

features list of int str pair of int pair of str The target features for which to create the

PDPs If featuresi is an int or a string a oneway PDP is created if featuresi is a tuple a

twoway PDP is created Each tuple must be of size 2 if any entry is a string then it must

be infeaturenames

featurenames seq of str shape nfeatures optional Name of each feature fea

turenamesi holds the name of the feature with index i By default the name of the feature

corresponds to their numerical index

target int optional defaultNone

• In a multiclass setting specifies the class for which the PDPs should be computed Note that for binary classification the positive class index 1 is always used

• In a multioutput setting specifies the task for which the PDPs should be computed

Ignored in binary classification or classical regression settings

responsemethod 'auto' 'predictproba' or 'decisionfunction' optional default'auto'

Specifies whether to use predictproba ordecisionfunction as the target response For

regressors this parameter is ignored and the response is always the output of predict By

633sklearninspection inspection 2229

scikitlearn user guide Release 0213

default predictproba is tried first and we revert to decisionfunction if it doesn't exist If method is 'recursion' the response is always the output of decisionfunction ncols int optional default3 The maximum number of columns in the grid plot gridresolution int optional default100 The number of equally spaced points on the axes of the plots for each target feature percentiles tuple of float optional default005 095 The lower and upper percentile used to create the extreme values for the PDP axes Must be in 0 1 method str optional default'auto' The method to use to calculate the partial dependence predictions

- 'recursion' is only supported for objects inheriting from BaseGradientBoosting but is more efficient in terms of speed With this method Xis optional and is only used to build the grid and the partial dependences are computed using the training data This method does not account for the init predictor of the boosting process which may lead to incorrect values see warning below With this method the target response of a classifier is always the decision function not the predicted probabilities
- 'brute' is supported for any estimator but is more computationally intensive
- If 'auto' then 'recursion' will be used for BaseGradientBoosting estimators with initNone and 'brute' for all other

Unlike the 'brute' method 'recursion' does not account for the init predictor of the boosting process In practice this still produces the same plots up to a constant offset in the target response

njobs int optional defaultNone The number of CPUs to use to compute the partial dependencesNone means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

verbose int optional default0 Verbose output during PD computations

figMatplotlib figure object optional defaultNone A figure object onto which the plots will be drawn after the figure has been cleared By default a new one is created

linekw dict optional Dict with keywords passed to the matplotlib.pyplot.plot call

For oneway partial dependence plots

contourkw dict optional Dict with keywords passed to the matplotlib.pyplot.plot call

For twoway partial dependence plots

Warning The 'recursion' method only works for gradient boosting estimators and unlike the 'brute' method it does not account for the init predictor of the boosting process In practice this will produce the same values as 'brute' up to a constant offset in the target response provided that init is a constant estimator which is the default However as soon as init is not a constant estimator the partial dependence values are incorrect for 'recursion'

See also

sklearn.inspection.partialdependence Return raw partial dependence values

2230 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

```
from sklearn.datasets import makefriedman1
from sklearn.ensemble import GradientBoostingRegressor
```

```
X, y = makefriedman1
```

```
clf = GradientBoostingRegressor(n_estimators=10)
clf.fit(X, y)
```

```
plot_partial_dependence(clf, X, [0, 1])
```

Examples using sklearn.inspection.plot_partial_dependence

• Partial Dependence Plots

634 sklearn.preprocessing: Preprocessing and Normalization

The sklearn.preprocessing module includes scaling, centering, normalization, binarization, and imputation methods.

User guide: See the Preprocessing data section for further details.

preprocessing.Binarizer: threshold. Copy binarize data set feature values to 0 or 1 according to a threshold.

preprocessing.FunctionTransformer: func

Constructs a transformer from an arbitrary callable.

preprocessing.KBinsDiscretizer: nbins

Bin continuous data into intervals.

preprocessing.KernelCenterer: Center a kernel matrix.

preprocessing.LabelBinarizer: neg_label

Binarize labels in a one-vs-all fashion.

preprocessing.LabelEncoder: Encode labels with value between 0 and n_classes-1.

preprocessing.MultiLabelBinarizer: classes

Transform between iterable of iterables and a multilabel.

format

preprocessing.MaxAbsScaler: copy. Scale each feature by its maximum absolute value.

preprocessing.MinMaxScaler: feature_range

copy. Transform features by scaling each feature to a given

range.

preprocessing.Normalizer: norm, copy. Normalize samples individually to unit norm.

preprocessing.OneHotEncoder: n_values. Encode categorical integer features as a one-hot numeric array.

preprocessing.OrdinalEncoder: categories

dtype. Encode categorical features as an integer array.

preprocessing.PolynomialFeatures: degree

Generate polynomial and interaction features.

preprocessing.PowerTransformer: method

Apply a power transform featurewise to make data more

Gaussian-like.

preprocessing.QuantileTransformer. Transform features using quantiles information.

preprocessing.RobustScaler: with_centering

Scale features using statistics that are robust to outliers.

preprocessing.StandardScaler: copy. Standardize features by removing the mean and scaling to unit variance.

634 sklearn.preprocessing: Preprocessing and Normalization 2231

scikitlearn user guide Release 0213

6341sklearnpreprocessing Binarizer

classsklearnpreprocessing Binarizer threshold00 copyTrue

Binarize data set feature values to 0 or 1 according to a threshold

Values greater than the threshold map to 1 while values less than or equal to the threshold map to 0 With the default threshold of 0 only positive values map to 1

Binarization is a common operation on text count data where the analyst can decide to only consider the presence or absence of a feature rather than a quantified number of occurrences for instance

It can also be used as a preprocessing step for estimators that consider boolean random variables eg modelled using the Bernoulli distribution in a Bayesian setting

Read more in the User Guide

Parameters

threshold float optional 00 by default Feature values below or equal to this are replaced by 0 above it by 1 Threshold may not be less than 0 for operations on sparse matrices

copy boolean optional default True set to False to perform inplace binarization and avoid a

copy if the input is already a numpy array or a scipysparse CSR (Compressed Sparse Row) matrix

See also

binarize Equivalent function without the estimator API (Application Programming Interface)

Notes

If the input is a sparse matrix only the nonzero values are subject to update by the Binarizer class

This estimator is stateless besides constructor parameters the fit method does nothing but is useful when used in a pipeline

Examples

from sklearnpreprocessing import Binarizer

X 1 1 2

2 0 0

0 1 1

transformer BinarizerfitX fit does nothing

transformer

BinarizercopyTrue threshold00

transformertransformX

array1 0 1

1 0 0

0 1 0

Methods

fitself X y Do nothing and return the estimator unchanged

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

Continued on next page

2232 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Table 6247 – continued from previous page

setparams self params Set the parameters of this estimator

transform self X copy Binarize each element of X

init selfthreshold00 copyTrue

fitselfXyNone

Do nothing and return the estimator unchanged

This method is just there to implement the usual API (Application Programming Interface) and hence work in pipelines

Parameters

Xarraylike

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it’s possible to update each component

of a nested object

Returns

self

transform selfXcopyNone

Binarize each element of X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The data to binarize element

by element scipysparse matrices should be in CSR (Compressed Sparse Row) format to avoid an unnecessary copy

copy bool Copy the input X or not

634sklearnpreprocessing Preprocessing and Normalization 2233

scikitlearn user guide Release 0213

6342sklearnpreprocessing FunctionTransformer

classsklearnpreprocessing FunctionTransformer funcNone inversefuncNone val

idateNone acceptsparseFalse

passy'deprecated'

checkinverseTrue kwargsNone

invkwargsNone

Constructs a transformer from an arbitrary callable

A FunctionTransformer forwards its X and optionally y arguments to a userdefined function or function

object and returns the result of this function This is useful for stateless transformations such as taking the log

of frequencies doing custom scaling etc

Note If a lambda is used as the function then the resulting transformer will not be pickleable

New in version 017

Read more in the User Guide

Parameters

func callable optional defaultNone The callable to use for the transformation This will be

passed the same arguments as transform with args and kwargs forwarded If func is None

then func will be the identity function

inversefunc callable optional defaultNone The callable to use for the inverse transforma

tion This will be passed the same arguments as inverse transform with args and kwargs

forwarded If inversefunc is None then inversefunc will be the identity function

validate bool optional defaultTrue Indicate that the input X array should be checked before

callingfunc The possibilities are

- If False there is no input validation

- If True then X will be converted to a 2dimensional NumPy array or sparse matrix If the

conversion is not possible an exception is raised

Deprecated since version 020 validateTrue as default will be replaced by

validateFalse in 022

acceptsparse boolean optional Indicate that func accepts a sparse matrix as input If validate

is False this has no effect Otherwise if acceptsparse is false sparse matrix inputs will

cause an exception to be raised

passy bool optional defaultFalse Indicate that transform should forward the y argument to

the inner callable

Deprecated since version 019

checkinverse bool defaultTrue Whether to check that or func followed by

inversefunc leads to the original inputs It can be used for a sanity check raising

a warning when the condition is not fulfilled

New in version 020

kwargs dict optional Dictionary of additional keyword arguments to pass to func

invkwargs dict optional Dictionary of additional keyword arguments to pass to in

versefunc

2234 Chapter 6 API (Application Programming Interface) Reference

Methods

fitself X y Fit transformer by checking X
fittransform self X y Fit to data then transform it
getparams self deep Get parameters for this estimator
inversetransform self X Transform X using the inverse function
setparams self params Set the parameters of this estimator
transform self X Transform X using the forward function
init self funcNone inversefuncNone validateNone acceptsparseFalse
passy'deprecated' checkinverseTrue kwargsNone invkwargsNone
fitselfXyNone
Fit transformer by checking X
Ifvalidate isTrue Xwill be checked
Parameters
Xarraylike shape nsamples nfeatures Input array
Returns
self
fittransform selfXyNone fitparams
Fit to data then transform it
Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X
Parameters
Xnumpy array of shape nsamples nfeatures Training set
ynumpy array of shape nsamples Target values
Returns
Xnew numpy array of shape nsamples nfeaturesnew Transformed array
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values
inversetransform selfX
Transform X using the inverse function
Parameters
Xarraylike shape nsamples nfeatures Input array
Returns
Xout arraylike shape nsamples nfeatures Transformed input
634sklearnpreprocessing Preprocessing and Normalization 2235

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Transform X using the forward function

Parameters

Xarraylike shape nsamples nfeatures Input array

Returns

Xout arraylike shape nsamples nfeatures Transformed input

Examples using sklearnpreprocessingFunctionTransformer

•Using FunctionTransformer to select columns

6343sklearnpreprocessing KBinsDiscretizer

classsklearnpreprocessing KBinsDiscretizer nbins5 encode'onehot' strat
egy'quantile'

Bin continuous data into intervals

Read more in the User Guide

Parameters

nbins int or arraylike shape nfeatures default5 The number of bins to produce

Raises ValueError if nbins 2

encode 'onehot' 'onehotdense' 'ordinal' default'onehot' Method used to encode the transformed result

onehot Encode the transformed result with onehot encoding and return a sparse matrix Ignored features are always stacked to the right

onehotdense Encode the transformed result with onehot encoding and return a dense array Ignored features are always stacked to the right

ordinal Return the bin identifier encoded as an integer value

strategy 'uniform' 'quantile' 'kmeans' default'quantile' Strategy used to define the widths of the bins

uniform All bins in each feature have identical widths

quantile All bins in each feature have the same number of points

kmeans Values in each bin have the same nearest center of a 1D kmeans cluster

Attributes

nbins int array shape nfeatures Number of bins per feature Bins whose width are too small ie 1e8 are removed with a warning

2236 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

binedges array of arrays shape nfeatures The edges of each bin Contain arrays of varying shapes nbins Ignored features will have empty arrays

See also

sklearnpreprocessingBinarizer class used to bin values as 0or1based on a parameter threshold

Notes

In bin edges for feature i the first and last values are used only for inversetransform During transform bin edges are extended to

npconcatenatenpinf binedgesi11 npinf

You can combine KBinsDiscretizer withsklearncomposeColumnTransformer if you only want to preprocess part of the features

KBinsDiscretizer might produce constant features eg when encode onehot and certain bins

do not contain any data These features can be removed with feature selection algorithms eg sklearn

featureselectionVarianceThreshold

Examples

X 2 1 4 1
1 2 3 05
0 3 2 05
1 4 1 2

est KBinsDiscretizernbins3 encodeordinal strategyuniform

estfitX

KBinsDiscretizer

Xt esttransformX

Xt

array 0 0 0 0
1 1 1 0
2 2 2 1
2 2 2 2

Sometimes it may be useful to convert the data back into the original feature space The inversetransform function converts the binned data into the original feature space Each value will be equal to the mean of the two bin edges

estbinedges0

array2 1 0 1

estinversetransformXt

array15 15 35 05
05 25 25 05
05 35 15 05
05 35 15 15

Methods

634sklearnpreprocessing Preprocessing and Normalization 2237

scikitlearn user guide Release 0213

fitself X y Fits the estimator

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

inversetransform self Xt Transforms discretized data back to original feature space

setparams self params Set the parameters of this estimator

transform self X Discretizes the data

init selfnbins5 encode'onehot' strategy'quantile'

fitselfXyNone

Fits the estimator

Parameters

Xnumeric arraylike shape nsamples nfeatures Data to be discretized

yignored

Returns

self

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfXt

Transforms discretized data back to original feature space

Note that this function does not regenerate the original data due to discretization rounding

Parameters

Xnumeric arraylike shape nsample nfeatures Transformed data in the binned space

Returns

Xinv numeric arraylike Data in the original feature space

2238 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Discretizes the data

Parameters

Xnumeric arraylike shape nsamples nfeatures Data to be discretized

Returns

Xnumeric arraylike or sparse matrix Data in the binned space

Examples using sklearnpreprocessingKBinsDiscretizer

- Using KBinsDiscretizer to discretize continuous features
- Demonstrating the different strategies of KBinsDiscretizer
- Feature discretization

6344sklearnpreprocessing KernelCenterer

classsklearnpreprocessing KernelCenterer

Center a kernel matrix

Let $K_{x,z}$ be a kernel defined by $\phi(x)^T \phi(z)$ where ϕ is a function mapping x to a Hilbert space

KernelCenterer centers ie normalize to have zero mean the data without explicitly computing $\phi(x)$ It is equivalent to centering $\phi(x)$ with sklearnpreprocessingStandardScalerwithstdFalse

Read more in the User Guide

Examples

```
from sklearnpreprocessing import KernelCenterer
from sklearnmetricspairwise import pairwise_kernels
```

```
X = [[1, 2, 2],
```

```
     [2, 1, 3],
```

```
     [4, 1, 2]]
```

```
K = pairwise_kernels(X, metric='linear')
```

```
K
```

```
array([[9, 2, 2],
```

```
       [2, 14, 13],
```

```
       [2, 13, 21]])
```

```
transformer = KernelCenterer().fit(K)
```

```
transformer
```

```
KernelCenterer
```

```
transformer.transform(K)
```

```
array([[5, 0, 5],
```

```
       [6, 3, 4],
```

```
       [6, 4, 3]])
```

scikitlearn user guide Release 0213

0 14 14

5 14 19

Methods

fitself K y Fit KernelCenterer

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self K copy Center kernel matrix

init self

fitselfKyNone

Fit KernelCenterer

Parameters

Knumpy array of shape nsamples nsamples Kernel matrix

Returns

self returns an instance of self

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

2240 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

transform selfKcopyTrue

Center kernel matrix

Parameters

Knumpy array of shape nsamples1 nsamples2 Kernel matrix

copy boolean optional default True Set to False to perform inplace computation

Returns

Knew numpy array of shape nsamples1 nsamples2

6345sklearnpreprocessing LabelBinarizer

classssklearnpreprocessing LabelBinarizer neglabel0 poslabel1

sparseoutputFalse

Binarize labels in a onevsall fashion

Several regression and binary classification algorithms are available in scikitlearn A simple way to extend

these algorithms to the multiclass classification case is to use the socalled onevsall scheme

At learning time this simply consists in learning one regressor or binary classifier per class In doing so one

needs to convert multiclass labels to binary labels belong or does not belong to the class LabelBinarizer

makes this process easy with the transform method

At prediction time one assigns the class for which the corresponding model gave the greatest confidence La

belBinarizer makes this easy with the inversetransform method

Read more in the User Guide

Parameters

neglabel int default 0 Value with which negative labels must be encoded

poslabel int default 1 Value with which positive labels must be encoded

sparseoutput boolean default False True if the returned array from transform is desired to

be in sparse CSR (Compressed Sparse Row) format

Attributes

classes array of shape nclass Holds the label for each class

ytype str Represents the type of the target data as evaluated by

utilsmulticlasstypeoftarget Possible type are ‘continuous’ ‘continuousmultioutput’

‘binary’ ‘multiclass’ ‘multiclassmultioutput’ ‘multilabelindicator’ and ‘unknown’

sparseinput boolean True if the input data to transform is given as a sparse matrix False

otherwise

See also

labelbinarize function to perform the transform operation of LabelBinarizer with fixed classes

sklearnpreprocessingOneHotEncoder encode categorical features using a onehot aka oneofK

scheme

634sklearnpreprocessing Preprocessing and Normalization 2241

scikitlearn user guide Release 0213

Examples

```
from sklearn import preprocessing
lb = preprocessing.LabelBinarizer
lb.fit([2, 6, 4, 2])
LabelBinarizer(neg_label=0, pos_label=1, sparse_output=False)
lb.classes_
array([2, 4, 6])
lb.transform([1, 6])
array([[1, 0, 0, 0],
       [0, 0, 0, 1]])
```

Binary targets transform to a column vector

```
lb = preprocessing.LabelBinarizer
lb.fit_transform([yes, no, no, yes])
array([0,
       0,
       1,
       1])
```

Passing a 2D matrix for multilabel classification

```
import numpy as np
lb.fit(np.array([[0, 1, 1, 1, 0, 0],
                 [1, 0, 0, 0, 1, 0]]))
LabelBinarizer(neg_label=0, pos_label=1, sparse_output=False)
lb.classes_
array([0, 1, 2])
lb.transform([[0, 1, 2, 1]])
array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1],
       [0, 1, 0]])
```

Methods

```
fit(self, y) Fit label binarizer
fit_transform(self, y) Fit label binarizer and transform multiclass labels to binary labels
get_params(self, deep=True) Get parameters for this estimator
inverse_transform(self, Y) Transform binary labels back to multiclass labels
set_params(self, **params) Set the parameters of this estimator
transform(self, y) Transform multiclass labels to binary labels
init(self, neg_label=0, pos_label=1, sparse_output=False)
fit(self, y)
```

Fit label binarizer

Parameters

y array of shape (n_samples, or n_samples, n_classes) Target values. The 2d matrix should only contain 0 and 1, represents multilabel classification.

Returns

2242 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

self returns an instance of self

fittransform selfy

Fit label binarizer and transform multiclass labels to binary labels

The output of transform is sometimes referred to as the 1ofK coding scheme

Parameters

yarray or sparse matrix of shape nsamples or nsamples nclasses Target values

The 2d matrix should only contain 0 and 1 represents multilabel classification Sparse

matrix can be CSR (Compressed Sparse Row) CSC COO DOK or LIL

Returns

Yarray or CSR (Compressed Sparse Row) matrix of shape nsamples nclasses Shape will be nsamples 1 for

binary problems

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfYthresholdNone

Transform binary labels back to multiclass labels

Parameters

Ynumpy array or sparse matrix with shape nsamples nclasses Target values All

sparse matrices are converted to CSR (Compressed Sparse Row) before inverse transformation

threshold float or None Threshold used in the binary and multilabel cases

Use 0 when Ycontains the output of decisionfunction classifier Use 05 when Ycon

tains the output of predictproba

If None the threshold is assumed to be half way between neglabel and poslabel

Returns

ynumpy array or CSR (Compressed Sparse Row) matrix of shape nsamples Target values

Notes

In the case when the binary labels are fractional probabilistic inversetransform chooses the class with

the greatest value Typically this allows to use the output of a linear model's decisionfunction method

directly as the input of inversetransform

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

634sklearnpreprocessing Preprocessing and Normalization 2243

scikitlearn user guide Release 0213

self

transform selfy

Transform multiclass labels to binary labels

The output of transform is sometimes referred to by some authors as the 1ofK coding scheme

Parameters

yarray or sparse matrix of shape nsamples or nsamples nclasses Target values

The 2d matrix should only contain 0 and 1 represents multilabel classification Sparse

matrix can be CSR (Compressed Sparse Row) CSC COO DOK or LIL

Returns

Ynumpy array or CSR (Compressed Sparse Row) matrix of shape nsamples nclasses Shape will be nsamples

1 for binary problems

6346sklearnpreprocessing LabelEncoder

classsklearnpreprocessing LabelEncoder

Encode labels with value between 0 and nclasses1

Read more in the User Guide

Attributes

classes array of shape nclass Holds the label for each class

See also

sklearnpreprocessingOrdinalEncoder encode categorical features using a onehot or ordinal

encoding scheme

Examples

LabelEncoder can be used to normalize labels

from sklearn import preprocessing

le_preprocessingLabelEncoder

lefit1 2 2 6

LabelEncoder

leclasses

array1 2 6

letransform1 1 2 6

array0 0 1 2

leinverttransform0 0 1 2

array1 1 2 6

It can also be used to transform nonnumerical labels as long as they are hashable and comparable to numerical labels

le_preprocessingLabelEncoder

lefitparis paris tokyo amsterdam

LabelEncoder

listleclasses

amsterdam paris tokyo

letransformtokyo tokyo paris

array2 2 1

2244 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

listleinverttransform2 2 1

tokyo tokyo paris

Methods

fitself y Fit label encoder

fittransform self y Fit label encoder and return encoded labels

getparams self deep Get parameters for this estimator

inverttransform self y Transform labels back to original encoding

setparams self params Set the parameters of this estimator

transform self y Transform labels to normalized encoding

init selfargs kwargs

Initialize self See helptypeself for accurate signature

fitselfy

Fit label encoder

Parameters

yarraylike of shape nsamples Target values

Returns

self returns an instance of self

fittransform selfy

Fit label encoder and return encoded labels

Parameters

yarraylike of shape nsamples Target values

Returns

yarraylike of shape nsamples

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inverttransform selfy

Transform labels back to original encoding

Parameters

ynumpy array of shape nsamples Target values

Returns

ynumpy array of shape nsamples

634sklearnpreprocessing Preprocessing and Normalization 2245

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfy

Transform labels to normalized encoding

Parameters

yarraylike of shape nsamples Target values

Returns

yarraylike of shape nsamples

6347sklearnpreprocessing MultiLabelBinarizer

classsklearnpreprocessing MultiLabelBinarizer classesNone sparseoutputFalse

Transform between iterable of iterables and a multilabel format

Although a list of sets or tuples is a very intuitive format for multilabel data it is unwieldy to process This transformer converts between this intuitive format and the supported multilabel format a samples x classes binary matrix indicating the presence of a class label

Parameters

classes arraylike of shape nclasses optional Indicates an ordering for the class labels All

entries should be unique cannot contain duplicate classes

sparseoutput boolean default False Set to true if output binary array is desired in CSR (Compressed Sparse Row)

sparse format

Attributes

classes array of labels A copy of the classes parameter where provided or otherwise the

sorted set of classes found when fitting

See also

sklearnpreprocessingOneHotEncoder encode categorical features using a onehot aka oneofK

scheme

Examples

from sklearnpreprocessing import MultiLabelBinarizer

mlb MultiLabelBinarizer

mlbfittransform1 2 3

array1 1 0

0 0 1

mlbclasses

array1 2 3

2246 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
mlbfittransformscifi thriller comedy
array0 1 1
1 0 0
listmlbclasses
comedy scifi thriller
Methods
fitself y Fit the label sets binarizer storing classes
fittransform self y Fit the label sets binarizer and transform the given label
sets
getparams self deep Get parameters for this estimator
inversetransform self yt Transform the given indicator matrix into label sets
setparams self params Set the parameters of this estimator
transform self y Transform the given label sets
init selfclassesNone sparseoutputFalse
fitselfy
Fit the label sets binarizer storing classes
Parameters
yiterable of iterables A set of labels any orderable and hashable object for each sample
If theclasses parameter is set ywill not be iterated
Returns
self returns this MultiLabelBinarizer instance
fittransform selfy
Fit the label sets binarizer and transform the given label sets
Parameters
yiterable of iterables A set of labels any orderable and hashable object for each sample
If theclasses parameter is set ywill not be iterated
Returns
yindicator array or CSR (Compressed Sparse Row) matrix shape nsamples nclasses A matrix such that
yindicatori j 1 iffclassesj is inyi and 0 otherwise
getparams selfdeepTrue
Get parameters for this estimator
Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators
Returns
params mapping of string to any Parameter names mapped to their values
inversetransform selfyt
Transform the given indicator matrix into label sets
Parameters
634sklearnpreprocessing Preprocessing and Normalization 2247

scikitlearn user guide Release 0213

ytarray or sparse matrix of shape nsamples nclasses A matrix containing only 1s and 0s

Returns

ylist of tuples The set of labels for each sample such that yi consists of classesj for eachyti j 1

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfy

Transform the given label sets

Parameters

yiterable of iterables A set of labels any orderable and hashable object for each sample

If theclasses parameter is set ywill not be iterated

Returns

yindicator array or CSR (Compressed Sparse Row) matrix shape nsamples nclasses A matrix such that yindicatori j 1 iffclassesj is inyi and 0 otherwise

6348sklearnpreprocessing MaxAbsScaler

classsklearnpreprocessing MaxAbsScaler copyTrue

Scale each feature by its maximum absolute value

This estimator scales and translates each feature individually such that the maximal absolute value of each feature in the training set will be 10 It does not shiftcenter the data and thus does not destroy any sparsity

This scaler can also be applied to sparse CSR (Compressed Sparse Row) or CSC matrices

New in version 017

Parameters

copy boolean optional default is True Set to False to perform inplace scaling and avoid a copy if the input is already a numpy array

Attributes

scale ndarray shape nfeatures Per feature relative scaling of the data

New in version 017 scale attribute

maxabs ndarray shape nfeatures Per feature maximum absolute value

nsamplesseen int The number of samples processed by the estimator Will be reset on new calls to fit but increments across partialfit calls

See also

maxabsscale Equivalent function without the estimator API (Application Programming Interface)

2248 Chapter 6 API (Application Programming Interface) Reference

Notes

NaNs are treated as missing values disregarded in fit and maintained in transform

For a comparison of the different scalers transformers and normalizers see exam

plespreprocessingplotallscalingpy

Examples

```
from sklearn.preprocessing import MaxAbsScaler
```

```
X = [[1, 1, 2],
```

```
     [2, 0, 0],
```

```
     [0, 1, 1]]
```

```
transformer = MaxAbsScaler()fit(X)
```

```
transformer
```

```
MaxAbsScaler(copy=True,
```

```
transformer.transform(X)
```

```
array([[0.5, 1., 1.],
```

```
       [1., 0., 0.],
```

```
       [0., 1., 0.5]])
```

Methods

fit(self, X, y) Compute the maximum absolute value to be used for

later scaling

fit_transform(self, X, y) Fit to data then transform it

get_params(self, deep=True) Get parameters for this estimator

inverse_transform(self, X) Scale back the data to the original representation

partial_fit(self, X, y) Online computation of max absolute value of X for later scaling

set_params(self, **kwargs) Set the parameters of this estimator

transform(self, X) Scale the data

init(self, copy=True)

fit(self, X, y=None)

Compute the maximum absolute value to be used for later scaling

Parameters

Xarraylike sparse matrix shape (n_samples, n_features) The data used to compute the perfeature minimum and maximum used for later scaling along the features axis

fit_transform(self, X, y=None, fit_params=None)

Fit to data then transform it

Fits transformer to X and y with optional parameters fit_params and returns a transformed version of X

Parameters

Xnumpy array of shape (n_samples, n_features) Training set

ynumpy array of shape (n_samples,) Target values

Returns

634sklearn.preprocessing Preprocessing and Normalization 2249

scikitlearn user guide Release 0213

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfX

Scale back the data to the original representation

Parameters

Xarraylike sparse matrix The data that should be transformed back

partialfit selfXyNone

Online computation of max absolute value of X for later scaling All of X is processed as a single batch

This is intended for cases when fit is not feasible due to very large number of nsamples or because X

is read from a continuous stream

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The data used to compute the

mean and standard deviation used for later scaling along the features axis

ylgnored

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Scale the data

Parameters

Xarraylike sparse matrix The data that should be scaled

Examples using sklearnpreprocessingMaxAbsScaler

•Compare the effect of different scalers on data with outliers

6349sklearnpreprocessing MinMaxScaler

classsklearnpreprocessing MinMaxScaler featurerange0 1copyTrue

Transforms features by scaling each feature to a given range

This estimator scales and translates each feature individually such that it is in the given range on the training set

eg between zero and one

2250 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The transformation is given by

$X_{std} = \frac{X - X_{minaxis0}}{X_{maxaxis0} - X_{minaxis0}}$

$X_{scaled} = X_{std} \cdot \max(\min, \min)$

where \min, \max = featurerange

The transformation is calculated as

$X_{scaled} = \text{scale} \cdot X - \min(X_{minaxis0}, \text{scale})$

where $\text{scale} = \max(\min, X_{maxaxis0} - X_{minaxis0})$

This transformation is often used as an alternative to zero mean unit variance scaling

Read more in the User Guide

Parameters

featurerange tuple min max default(0, 1) Desired range of transformed data

copy boolean optional default True Set to False to perform inplace row normalization and

avoid a copy if the input is already a numpy array

Attributes

min ndarray shape nfeatures Per feature adjustment for minimum Equivalent to min

$X_{minaxis0}$ selfscale

scale ndarray shape nfeatures Per feature relative scaling of the data Equivalent to max

$\min(X_{maxaxis0}, X_{minaxis0})$

New in version 0.17 scale attribute

datamin ndarray shape nfeatures Per feature minimum seen in the data

New in version 0.17 datamin

datamax ndarray shape nfeatures Per feature maximum seen in the data

New in version 0.17 datamax

datarange ndarray shape nfeatures Per feature range datamax

datamin seen in the data

New in version 0.17 datarange

See also

minmaxscale Equivalent function without the estimator API (Application Programming Interface)

Notes

NaNs are treated as missing values disregarded in fit and maintained in transform

For a comparison of the different scalers transformers and normalizers see exam

plespreprocessingplotallscalingpy

Examples

634sklearnpreprocessing Preprocessing and Normalization 2251

```
scikitlearn user guide Release 0213
from sklearn.preprocessing import MinMaxScaler
data = 1 2 05 6 0 10 1 18
scaler = MinMaxScaler
print(scaler.fit(data))
MinMaxScaler(copy=True, feature_range=(0, 1))
print(scaler.data_max_)
1 18
print(scaler.transform(data))
0 0
0.25 0.25
0.5 0.5
1 1
print(scaler.transform2(data))
15 0
Methods
fit(self, X, y) Compute the minimum and maximum to be used for
later scaling
fit_transform(self, X, y) Fit to data then transform it
get_params(self, deep=True) Get parameters for this estimator
inverse_transform(self, X) Undo the scaling of X according to feature_range
partial_fit(self, X, y) Online computation of min and max on X for later scal
ing
set_params(self, **kwargs) Set the parameters of this estimator
transform(self, X) Scaling features of X according to feature_range
init(self, feature_range=(0, 1), copy=True)
fit(self, X, y=None)
Compute the minimum and maximum to be used for later scaling
Parameters
X array-like shape (n_samples, n_features) The data used to compute the perfeature min
imum and maximum used for later scaling along the features axis
fit_transform(self, X, y=None, fit_params=None)
Fit to data then transform it
Fits transformer to X and y with optional parameters fit_params and returns a transformed version of X
Parameters
X numpy array of shape (n_samples, n_features) Training set
y numpy array of shape (n_samples,) Target values
Returns
X_new numpy array of shape (n_samples, n_features) Transformed array
get_params(self, deep=True)
Get parameters for this estimator
Parameters
2252 Chapter 6 API (Application Programming Interface) Reference
```

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfX

Undo the scaling of X according to featurerange

Parameters

Xarraylike shape nsamples nfeatures Input data that will be transformed It cannot be sparse

partialfit selfXyNone

Online computation of min and max on X for later scaling All of X is processed as a single batch This is intended for cases when fit is not feasible due to very large number of nsamples or because X is read from a continuous stream

Parameters

Xarraylike shape nsamples nfeatures The data used to compute the mean and standard deviation used for later scaling along the features axis

yIgnored

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Scaling features of X according to featurerange

Parameters

Xarraylike shape nsamples nfeatures Input data that will be transformed

Examples using sklearnpreprocessingMinMaxScaler

- Compare Stochastic learning strategies for MLPClassifier
- Compare the effect of different scalers on data with outliers

63410sklearnpreprocessing Normalizer

classsklearnpreprocessing Normalizer norm'12' copyTrue

Normalize samples individually to unit norm

Each sample ie each row of the data matrix with at least one non zero component is rescaled independently of other samples so that its norm l1 or l2 equals one

This transformer is able to work both with dense numpy arrays and scipysparse matrix use CSR (Compressed Sparse Row) format if you want to avoid the burden of a copy conversion

634sklearnpreprocessing Preprocessing and Normalization 2253

scikitlearn user guide Release 0213

Scaling inputs to unit norms is a common operation for text classification or clustering for instance For instance the dot product of two l2normalized TFIDF vectors is the cosine similarity of the vectors and is the base similarity metric for the Vector Space Model commonly used by the Information Retrieval community

Read more in the User Guide

Parameters

norm 'l1' 'l2' or 'max' optional 'l2' by default The norm to use to normalize each non

zero sample

copy boolean optional default True set to False to perform inplace row normalization and

avoid a copy if the input is already a numpy array or a scipysparse CSR (Compressed Sparse Row) matrix

See also

normalize Equivalent function without the estimator API (Application Programming Interface)

Notes

This estimator is stateless besides constructor parameters the fit method does nothing but is useful when used in a pipeline

For a comparison of the different scalers transformers and normalizers see exam

plespreprocessingplotallscalingpy

Examples

from sklearnpreprocessing import Normalizer

X 4 1 2 2

1 3 9 3

5 7 5 1

transformer NormalizerfitX fit does nothing

transformer

NormalizercopyTrue norml2

transformertransformX

array08 02 04 04

01 03 09 03

05 07 05 01

Methods

fitself X y Do nothing and return the estimator unchanged

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X copy Scale each non zero row of X to unit norm

init selfnorm'l2' copyTrue

fitselfXyNone

Do nothing and return the estimator unchanged

This method is just there to implement the usual API (Application Programming Interface) and hence work in pipelines

2254 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

Xarraylike

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfXcopyNone

Scale each non zero row of X to unit norm

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The data to normalize row

by row scipysparse matrices should be in CSR (Compressed Sparse Row) format to avoid an unnecessary copy

copy bool optional default None Copy the input X or not

Examples using sklearnpreprocessingNormalizer

•Compare the effect of different scalers on data with outliers

•Clustering text documents using kmeans

634sklearnpreprocessing Preprocessing and Normalization 2255

Encode categorical integer features as a onehot numeric array

The input to this transformer should be an arraylike of integers or strings denoting the values taken on by

categorical discrete features The features are encoded using a onehot aka 'oneofK' or 'dummy' encoding

scheme This creates a binary column for each category and returns a sparse matrix or dense array

By default the encoder derives the categories based on the unique values in each feature Alternatively you can

also specify the categories manually The OneHotEncoder previously assumed that the input features take

on values in the range 0 maxvalues This behaviour is deprecated

This encoding is needed for feeding categorical data to many scikitlearn estimators notably linear models and

SVMs with the standard kernels

Note a onehot encoding of y labels should use a LabelBinarizer instead

Read more in the User Guide

Parameters

categories 'auto' or a list of listsarrays of values default'auto' Categories unique values

per feature

- 'auto' Determine categories automatically from the training data
- list categoriesi holds the categories expected in the ith column The passed categories should not mix strings and numeric values within a single feature and should be sorted in case of numeric values

The used categories can be found in the categories attribute

drop 'first' or a listarray of shape nfeatures defaultNone Specifies a methodology to

use to drop one of the categories per feature This is useful in situations where perfectly

collinear features cause problems such as when feeding the resulting data into a neural

network or an unregularized regression

- None retain all features the default
- 'first' drop the first category in each feature If only one category is present the feature will be dropped entirely
- array dropi is the category in feature X i that should be dropped

sparse boolean defaultTrue Will return sparse matrix if set True else will return an array

dtype number type defaultnpfloat Desired dtype of output

handleunknown 'error' or 'ignore' default'error' Whether to raise an error or ignore if

an unknown categorical feature is present during transform default is to raise When this

parameter is set to 'ignore' and an unknown category is encountered during transform the

resulting onehot encoded columns for this feature will be all zeros In the inverse transform

an unknown category will be denoted as None

nvalues 'auto' int or array of ints default'auto' Number of values per feature

- 'auto' determine value range from training data
- inthnumber of categorical values per feature Each feature value should be in rangenvalues

scikitlearn user guide Release 0213

•array nvaluesi is the number of categorical values in X i Each feature value should be in rangenvaluesi

Deprecated since version 020 The nvalues keyword was deprecated in version 020 and will be removed in 022 Use categories instead

categoricalfeatures ‘all’ or array of indices or mask default‘all’ Specify what features are treated as categorical

- ‘all’ All features are treated as categorical
- array of indices Array of categorical feature indices
- mask Array of length nfeatures and with dtypebool

Noncategorical features are always stacked to the right of the matrix

Deprecated since version 020 The categoricalfeatures keyword was deprecated in version 020 and will be removed in 022 You can use the ColumnTransformer instead

Attributes

categories list of arrays The categories of each feature determined during fitting in order of the features in X and corresponding with the output of transform This includes the category specified in drop if any

dropidx array of shape nfeatures dropidxi is the index in categoriesi of the category to be dropped for each feature None if all the transformed features will be retained

activefeatures array Indices for active features meaning values that actually occur in the training set Only available when nvalues is auto

Deprecated since version 020 The activefeatures attribute was deprecated in version 020 and will be removed in 022

featureindices array of shape nfeatures Indices to feature ranges Feature iin the original data is mapped to features from featureindicesi to featureindicesi1 and then potentially masked by activefeatures af

terwards
Deprecated since version 020 The featureindices attribute was deprecated in version 020 and will be removed in 022

nvalues array of shape nfeatures Maximum number of values per feature

Deprecated since version 020 The nvalues attribute was deprecated in version 020 and will be removed in 022

See also

sklearnpreprocessingOrdinalEncoder performs an ordinal integer encoding of the categorical features

sklearnfeatureextractionDictVectorizer performs a onehot encoding of dictionary items also handles stringvalued features

sklearnfeatureextractionFeatureHasher performs an approximate onehot encoding of dictionary items or strings

sklearnpreprocessingLabelBinarizer binarizes labels in a onevsall fashion

634sklearnpreprocessing Preprocessing and Normalization 2257

scikitlearn user guide Release 0213

sklearnpreprocessingMultiLabelBinarizer transforms between iterable of iterables and a multilabel format eg a samples x classes binary matrix indicating the presence of a class label

Examples

Given a dataset with two features we let the encoder find the unique values per feature and transform the data to a binary onehot encoding

```
from sklearnpreprocessing import OneHotEncoder
enc = OneHotEncoder(handleunknownignore
X = Male 1 Female 3 Female 2
encfitX
```

OneHotEncodercategoricalfeaturesNone categoriesNone dropNone

dtype numpyfloat64 handleunknownignore

nvaluesNone sparseTrue

enccategories

arrayFemale Male dtypeobject array1 2 3 dtypeobject

encctransformFemale 1 Male 4toarray

array1 0 1 0 0

0 1 0 0 0

encinversetransform0 1 1 0 0 0 0 0 1 0

arrayMale 1

None 2 dtypeobject

encgetfeaturenames

arrayx0Female x0Male x11 x12 x13 dtypeobject

dropenc = OneHotEncoderdropfirstfitX

dropenccategories

arrayFemale Male dtypeobject array1 2 3 dtypeobject

dropencctransformFemale 1 Male 2toarray

array0 0 0

1 1 0

Methods

fitself X y Fit OneHotEncoder to X

fittransform self X y Fit OneHotEncoder to X then transform X

getfeaturenames self inputfeatures Return feature names for output features

getparams self deep Get parameters for this estimator

inversetransform self X Convert the back data to the original representation

setparams self params Set the parameters of this estimator

transform self X Transform X using onehot encoding

init selfnvaluesNone categoricalfeaturesNone categoriesNone dropNone

sparseTrue dtypeclass 'numpyfloat64' handleunknown'error'

fitselfXyNone

Fit OneHotEncoder to X

Parameters

Xarraylike shape nsamples nfeatures The data to determine the categories of each

2258 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

feature

Returns

self

fittransform selfXyNone

Fit OneHotEncoder to X then transform X

Equivalent to fitXtransformX but more convenient

Parameters

Xarraylike shape nsamples nfeatures The data to encode

Returns

Xout sparse matrix if sparseTrue else a 2d array Transformed input

getfeaturenames selfinputfeaturesNone

Return feature names for output features

Parameters

inputfeatures list of string length nfeatures optional String names for input features if

available By default "x0" "x1" "xnfeatures" is used

Returns

outputfeaturenames array of string length noutputfeatures

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfX

Convert the back data to the original representation

In case unknown categories are encountered all zeros in the onehot encoding None is used to represent

this category

Parameters

Xarraylike or sparse matrix shape nsamples nencodedfeatures The transformed

data

Returns

Xtr arraylike shape nsamples nfeatures Inverse transformed array

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

634sklearnpreprocessing Preprocessing and Normalization 2259

scikitlearn user guide Release 0213

transform selfX

Transform X using onehot encoding

Parameters

Xarraylike shape nsamples nfeatures The data to encode

Returns

Xout sparse matrix if sparseTrue else a 2d array Transformed input

Examples using sklearnpreprocessingOneHotEncoder

- Column Transformer with Mixed Types

- Feature transformations with ensembles of trees

63412sklearnpreprocessing OrdinalEncoder

classsklearnpreprocessing OrdinalEncoder categories'auto' dtypeclass

'numpyfloat64'

Encode categorical features as an integer array

The input to this transformer should be an arraylike of integers or strings denoting the values taken on by categorical discrete features The features are converted to ordinal integers This results in a single column of integers 0 to ncategories - 1 per feature

Read more in the User Guide

Parameters

categories 'auto' or a list of listsarrays of values Categories unique values per feature

- 'auto' Determine categories automatically from the training data

- list categoriesi holds the categories expected in the ith column The passed

categories should not mix strings and numeric values and should be sorted in case of numeric values

The used categories can be found in the categories attribute

dtype number type default npfloat64 Desired dtype of output

Attributes

categories list of arrays The categories of each feature determined during fitting in order of the features in X and corresponding with the output of transform

See also

sklearnpreprocessingOneHotEncoder performs a onehot encoding of categorical features

sklearnpreprocessingLabelEncoder encodes target labels with values between 0 and nclasses

1

Examples

Given a dataset with two features we let the encoder find the unique values per feature and transform the data to an ordinal encoding

2260 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

```
from sklearn.preprocessing import OrdinalEncoder
enc = OrdinalEncoder
X = Male 1 Female 3 Female 2
enc.fit(X)
```

```
OrdinalEncoder(categories=auto dtype numpyfloat64
enc.categories
array(Female Male dtypeobject array(1 2 3 dtypeobject
enc.transform(Female 3 Male 1
array(0 2
1 0
enc.inverse_transform(1 0 0 1
```

```
array(Male 1
Female 2 dtypeobject
Methods
fit(self X y) Fit the OrdinalEncoder to X
fit_transform(self X y) Fit to data then transform it
get_params(self deep) Get parameters for this estimator
inverse_transform(self X) Convert the data back to the original representation
set_params(self params) Set the parameters of this estimator
transform(self X) Transform X to ordinal codes
init(self categories='auto' dtypeclass 'numpyfloat64'
fit(self X y None)
```

```
Fit the OrdinalEncoder to X
Parameters
X arraylike shape nsamples nfeatures The data to determine the categories of each
feature
Returns
self
```

```
fit_transform(self X y None fit_params
Fit to data then transform it
Fits transformer to X and y with optional parameters fit_params and returns a transformed version of X
Parameters
X numpy array of shape nsamples nfeatures Training set
y numpy array of shape nsamples Target values
Returns
X_new numpy array of shape nsamples nfeatures_new Transformed array
```

```
get_params(self deep=True
Get parameters for this estimator
Parameters
```

scikitlearn user guide Release 0213

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfX

Convert the data back to the original representation

Parameters

Xarraylike or sparse matrix shape nsamples nencodedfeatures The transformed data

Returns

Xtr arraylike shape nsamples nfeatures Inverse transformed array

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

transform selfX

Transform X to ordinal codes

Parameters

Xarraylike shape nsamples nfeatures The data to encode

Returns

Xout sparse matrix or a 2d array Transformed input

63413sklearnpreprocessing PolynomialFeatures

classssklearnpreprocessing PolynomialFeatures degree2 interactiononlyFalse in cludebiasTrue order'C'

Generate polynomial and interaction features

Generate a new feature matrix consisting of all polynomial combinations of the features with degree less than or equal to the specified degree For example if an input sample is two dimensional and of the form a b the

degree2 polynomial features are 1 a b a2 ab b2

Parameters

degree integer The degree of the polynomial features Default 2

interactiononly boolean default False If true only interaction features are produced features that are products of at most degree distinct input features so not x12

x0x23 etc

includebias boolean If True default then include a bias column the feature in which all polynomial powers are zero ie a column of ones acts as an intercept term in a linear model

2262 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
order str in 'C' 'F' default 'C' Order of output array in the dense case 'F' order is faster
to compute but may slow down subsequent estimators
New in version 021

Attributes
powers array shape noutputfeatures ninputfeatures powers[i][j] is the exponent of
the jth input in the ith output
ninputfeatures int The total number of input features
noutputfeatures int The total number of polynomial output features The number of out
put features is computed by iterating over all suitably sized combinations of input features

Notes
Be aware that the number of features in the output array scales polynomially in the number of features of the
input array and exponentially in the degree High degrees can cause overfitting
Seeexampleslinearmodelplotpolynomialinterpolationpy

Examples
import numpy as np
from sklearnpreprocessing import PolynomialFeatures
X np.arange(6).reshape(3, 2)

X
array0 1
2 3
4 5
poly PolynomialFeatures2
polyfittransformX
array 1 0 1 0 0 1
1 2 3 4 6 9
1 4 5 16 20 25
poly PolynomialFeaturesinteractiononly True
polyfittransformX
array 1 0 1 0
1 2 3 6
1 4 5 20

Methods
fitself X y Compute number of output features
fittransform self X y Fit to data then transform it
getfeaturenames self inputfeatures Return feature names for output features
getparams self deep Get parameters for this estimator
setparams self params Set the parameters of this estimator
transform self X Transform data to polynomial features
init selfdegree2 interactiononlyFalse includebiasTrue order'C'
fitselfXyNone
634sklearnpreprocessing Preprocessing and Normalization 2263

scikitlearn user guide Release 0213

Compute number of output features

Parameters

Xarraylike shape nsamples nfeatures The data

Returns

self instance

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getfeaturenames selfinputfeaturesNone

Return feature names for output features

Parameters

inputfeatures list of string length nfeatures optional String names for input features if available By default “x0” “x1” “xnfeatures” is used

Returns

outputfeaturenames list of string length noutputfeatures

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

transform selfX

Transform data to polynomial features

Parameters

Xarraylike or CSR CSC sparse matrix shape nsamples nfeatures The data to transform row by row

2264 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Prefer CSR (Compressed Sparse Row) over CSC for sparse input for speed but CSC is required if the degree is 4 or higher. If the degree is less than 4 and the input format is CSC it will be converted to CSR (Compressed Sparse Row). Polynomial features generated then converted back to CSC. If the degree is 2 or 3 the method described in “Leveraging Sparsity to Speed Up Polynomial Feature Expansions of CSR (Compressed Sparse Row) Matrices Using KSimplex Numbers” by Andrew Nysrom and John Hughes is used which is much faster than the method used on CSC input. For this reason a CSC input will be converted to CSR (Compressed Sparse Row) and the output will be converted back to CSC prior to being returned hence the preference of CSR (Compressed Sparse Row).

Returns
X: np.ndarray or CSR/CSC sparse matrix shape (n_samples, n_polynomial_features)
The matrix of features where n_polynomial_features is the number of polynomial features generated from the combination of inputs.
Examples using sklearn.preprocessing.PolynomialFeatures

- Polynomial interpolation
- Robust linear estimator fitting
- Underfitting vs Overfitting

63414 sklearn.preprocessing.PowerTransformer
class sklearn.preprocessing.PowerTransformer method 'yeo-johnson' standardize=True
copy=True

Apply a power transform featurewise to make data more Gaussian-like. Power transforms are a family of parametric monotonic transformations that are applied to make data more Gaussian-like. This is useful for modeling issues related to heteroscedasticity, nonconstant variance, or other situations where normality is desired. Currently, PowerTransformer supports the Box-Cox transform and the Yeo-Johnson transform. The optimal parameter for stabilizing variance and minimizing skewness is estimated through maximum likelihood. Box-Cox requires input data to be strictly positive, while Yeo-Johnson supports both positive or negative data. By default, zero-mean unit-variance normalization is applied to the transformed data. Read more in the User Guide.

Parameters
method: str, default='yeo-johnson' The power transform method. Available methods are:
• 'yeo-johnson' works with positive and negative values
• 'box-cox' only works with strictly positive values
standardize: boolean, default=True Set to True to apply zero-mean unit-variance normalization to the transformed output.
copy: boolean, optional, default=True Set to False to perform inplace computation during transformation.

Attributes
lambda_: array of float shape (n_features,) The parameters of the power transformation for the selected features.

See also
634 sklearn.preprocessing.Preprocessing and Normalization 2265

scikitlearn user guide Release 0213

powertransform Equivalent function without the estimator API (Application Programming Interface)

QuantileTransformer Maps data to a standard normal distribution with the parameter

outputdistributionnormal

Notes

NaNs are treated as missing values disregarded in fit and maintained in transform

For a comparison of the different scalers transformers and normalizers see exam

plespreprocessingplotallscalingpy

References

Rf3e1504535de1 Rf3e1504535de2

Examples

```
import numpy as np
from sklearn.preprocessing import PowerTransformer
pt = PowerTransformer
data = [1, 2, 3, 2, 4, 5]
print(pt.fit(data))
PowerTransformer(copy=True, method='yeojohnson', standardize=True)
print(pt.lambda_)
1386 3100
print(pt.transform(data))
1316 0707
0209 0707
1106 1414
```

Methods

fit(self, X, y) Estimate the optimal parameter lambda for each feature

fit_transform(self, X, y)

get_params(self, deep=True) Get parameters for this estimator

inverse_transform(self, X) Apply the inverse power transformation using the fitted

lambda_

set_params(self, **kwargs) Set the parameters of this estimator

transform(self, X) Apply the power transform to each feature using the fit

ted lambda_

init(self, method='yeojohnson', standardize=True, copy=True)

fit(self, X, y=None)

Estimate the optimal parameter lambda for each feature

The optimal lambda parameter for minimizing skewness is estimated on each feature independently using

maximum likelihood

Parameters

2266 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures The data used to estimate the optimal transformation parameters

ylgnored

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfX

Apply the inverse power transformation using the fitted lambdas

The inverse of the BoxCox transformation is given by

if lambda 0

X expXtrans

else

X Xtrans lambda 1 lambda

The inverse of the YeoJohnson transformation is given by

ifX 0and lambda 0

X expXtrans 1

elifX 0and lambda 0

X Xtrans lambda 1 lambda 1

elifX 0and lambda 2

X 1 2 lambdaXtrans 1 1 2 lambda

elifX 0and lambda 2

X 1 expXtrans

Parameters

Xarraylike shape nsamples nfeatures The transformed data

Returns

Xarraylike shape nsamples nfeatures The original data

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Apply the power transform to each feature using the fitted lambdas

634sklearnpreprocessing Preprocessing and Normalization 2267

scikitlearn user guide Release 0213

Parameters

Xarraylike shape nsamples nfeatures The data to be transformed using a power transformation

Returns

Xtrans arraylike shape nsamples nfeatures The transformed data

Examples using sklearnpreprocessingPowerTransformer

- Map data to a normal distribution
- Compare the effect of different scalers on data with outliers

63415sklearnpreprocessing QuantileTransformer

classsklearnpreprocessing QuantileTransformer nquantiles1000 out

putdistribution'uniform' ig

noreimplicitzerosFalse subsam

ple100000 randomstateNone

copyTrue

Transform features using quantiles information

This method transforms the features to follow a uniform or a normal distribution Therefore for a given feature this transformation tends to spread out the most frequent values It also reduces the impact of marginal outliers this is therefore a robust preprocessing scheme

The transformation is applied on each feature independently First an estimate of the cumulative distribution function of a feature is used to map the original values to a uniform distribution The obtained values are then mapped to the desired output distribution using the associated quantile function Features values of newunseen data that fall below or above the fitted range will be mapped to the bounds of the output distribution Note that this transform is nonlinear It may distort linear correlations between variables measured at the same scale but renders variables measured at different scales more directly comparable

Read more in the User Guide

Parameters

nquantiles int optional default1000 or nsamples Number of quantiles to be computed

It corresponds to the number of landmarks used to discretize the cumulative distribution

function If nquantiles is larger than the number of samples nquantiles is set to the number of samples as a larger number of quantiles does not give a better approximation of the cumulative distribution function estimator

outputdistribution str optional default'uniform' Marginal distribution for the trans

formed data The choices are 'uniform' default or 'normal'

ignoreimplicitzeros bool optional defaultFalse Only applies to sparse matrices If True the sparse entries of the matrix are discarded to compute the quantile statistics If False these entries are treated as zeros

subsample int optional default1e5 Maximum number of samples used to estimate the quantiles for computational efficiency Note that the subsampling procedure may differ for valueidentical sparse and dense matrices

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

2268 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

the RandomState instance used by nprandom Note that this is used by subsampling and smoothing noise

copy boolean optional defaultTrue Set to False to perform inplace transformation and avoid a copy if the input is already a numpy array

Attributes

nquantiles integer The actual number of quantiles used to discretize the cumulative distribution function

quantiles ndarray shape nquantiles nfeatures The values corresponding the quantiles of reference

references ndarray shapenquantiles Quantiles of references

See also

quantiletransform Equivalent function without the estimator API (Application Programming Interface)

PowerTransformer Perform mapping to a normal distribution using a power transform

StandardScaler Perform standardization that is faster but less robust to outliers

RobustScaler Perform robust standardization that removes the influence of outliers but does not put outliers and inliers on the same scale

Notes

NaNs are treated as missing values disregarded in fit and maintained in transform

For a comparison of the different scalers transformers and normalizers see examplespreprocessingplotsallscalingpy

Examples

```
import numpy as np
from sklearn.preprocessing import QuantileTransformer
rng = np.random.RandomState(0)
X = np.sort(rng.normal(loc=0.5, scale=0.25, size=(25, 1)), axis=0)
qt = QuantileTransformer(nquantiles=10, random_state=0)
qt.fit_transform(X)
array
```

Methods

fitself X y Compute the quantiles used for transforming

fit_transformself X y Fit to data then transform it

get_paramsself deep Get parameters for this estimator

inverse_transformself X Backprojection to the original space

set_paramsself params Set the parameters of this estimator

transformself X Featurewise transformation of the data

634sklearnpreprocessing Preprocessing and Normalization 2269

scikitlearn user guide Release 0213

init selfnquantiles1000 outputdistribution'uniform' ignoreimplicitzerosFalse sub
sample100000 randomstateNone copyTrue
fitselfXyNone

Compute the quantiles used for transforming

Parameters

Xndarray or sparse matrix shape nsamples nfeatures The data used to scale
along the features axis If a sparse matrix is provided it will be converted into
a sparsecscmatrix Additionally the sparse matrix needs to be nonnegative if
ignoreimplicitzeros is False

Returns

self object

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set
ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfX

Backprojection to the original space

Parameters

Xndarray or sparse matrix shape nsamples nfeatures The data used to scale
along the features axis If a sparse matrix is provided it will be converted into
a sparsecscmatrix Additionally the sparse matrix needs to be nonnegative if
ignoreimplicitzeros is False

Returns

Xndarray or sparse matrix shape nsamples nfeatures The projected data

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have
parameters of the form componentparameter so that it's possible to update each component
of a nested object

Returns

2270 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

self
transform selfX

Featurewise transformation of the data

Parameters

Xndarray or sparse matrix shape nsamples nfeatures The data used to scale along the features axis If a sparse matrix is provided it will be converted into a sparsecscmatrix Additionally the sparse matrix needs to be nonnegative if ignoreimplicitzeros is False

Returns

Xndarray or sparse matrix shape nsamples nfeatures The projected data

Examples using sklearnpreprocessingQuantileTransformer

- Effect of transforming the targets in regression model
- Map data to a normal distribution
- Compare the effect of different scalers on data with outliers

63416sklearnpreprocessing RobustScaler

classsklearnpreprocessing RobustScaler withcenteringTrue withscalingTrue quantilerange250 750 copyTrue

Scale features using statistics that are robust to outliers

This Scaler removes the median and scales the data according to the quantile range defaults to IQR Interquartile Range The IQR is the range between the 1st quartile 25th quantile and the 3rd quartile 75th quantile

Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set Median and interquartile range are then stored to be used on later data using the transform method

Standardization of a dataset is a common requirement for many machine learning estimators Typically this is done by removing the mean and scaling to unit variance However outliers can often influence the sample mean variance in a negative way In such cases the median and the interquartile range often give better results

New in version 017

Read more in the User Guide

Parameters

withcentering boolean True by default If True center the data before scaling This will causetransform to raise an exception when attempted on sparse matrices because centering them entails building a dense matrix which in common use cases is likely to be too large to fit in memory

withscaling boolean True by default If True scale the data to interquartile range

quantilerange tuple qmin qmax 00 qmin qmax 1000 Default 250 750

1st quartile 3rd quartile IQR Quantile range used to calculate scale

New in version 018

copy boolean optional default is True If False try to avoid a copy and do inplace scaling instead This is not guaranteed to always work inplace eg if the data is not a NumPy array or scipysparse CSR (Compressed Sparse Row) matrix a copy may still be returned

634sklearnpreprocessing Preprocessing and Normalization 2271

scikitlearn user guide Release 0213

Attributes

center array of floats The median value for each feature in the training set

scale array of floats The scaled interquartile range for each feature in the training set

New in version 0.17: scale attribute

See also

robustscale Equivalent function without the estimator API (Application Programming Interface)

sklearn.decomposition.PCA Further removes the linear correlation across features with

whiten=True

Notes

For a comparison of the different scalers, transformers and normalizers see exam

ples/preprocessing/plot_all_scaling.py

<https://en.wikipedia.org/wiki/Median> https://en.wikipedia.org/wiki/Interquartile_range

Examples

```
from sklearn.preprocessing import RobustScaler
```

```
X = [[1, 2, 2],
```

```
     [2, 1, 3],
```

```
     [4, 1, 2]]
```

```
transformer = RobustScaler().fit(X)
```

```
transformer
```

```
RobustScaler(copy=True, quantile_range=(250, 750), with_centering=True,
```

```
with_scaling=True)
```

```
transformer.transform(X)
```

```
array([[0.2, 0.],
```

```
       [1.0, 0.04],
```

```
       [1.0, 0.16]])
```

Methods

fit(self, X, y) Compute the median and quantiles to be used for scal

ing

fit_transform(self, X, y) Fit to data then transform it

get_params(self, deep=True) Get parameters for this estimator

inverse_transform(self, X) Scale back the data to the original representation

set_params(self, **params) Set the parameters of this estimator

transform(self, X) Center and scale the data

__init__(self, with_centering=True, with_scaling=True, quantile_range=(250, 750), copy=True)

fit(self, X, y=None)

Compute the median and quantiles to be used for scaling

Parameters

2272 Chapter 6: API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures The data used to compute the median and quantiles used for later scaling along the features axis

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfX

Scale back the data to the original representation

Parameters

Xarraylike The data used to scale along the specified axis

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Center and scale the data

Parameters

Xarraylike sparse matrix The data used to scale along the specified axis

Examples using sklearnpreprocessingRobustScaler

•Compare the effect of different scalers on data with outliers

63417sklearnpreprocessing StandardScaler

classssklearnpreprocessing StandardScaler copyTrue withmeanTrue withstdTrue

Standardize features by removing the mean and scaling to unit variance

634sklearnpreprocessing Preprocessing and Normalization 2273

scikitlearn user guide Release 0213

The standard score of a sample x_i is calculated as

$$z = \frac{x_i - \mu}{\sigma}$$

where μ is the mean of the training samples or zero if `withmean=False` and σ is the standard deviation of the training samples or one if `withstd=False`

Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using the `transform` method.

Standardization of a dataset is a common requirement for many machine learning estimators; they might behave badly if the individual features do not more or less look like standard normally distributed data (e.g. Gaussian) with 0 mean and unit variance.

For instance, many elements used in the objective function of a learning algorithm (such as the RBF (Radial Basis Function) kernel of Support Vector Machines or the L1 and L2 regularizers of linear models) assume that all features are centered around 0 and have variance in the same order. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

This scaler can also be applied to sparse CSR (Compressed Sparse Row) or CSC matrices by passing `withmean=False` to avoid breaking the sparsity structure of the data.

Read more in the User Guide

Parameters

`copy` boolean, optional, default: True. If False, try to avoid a copy and do inplace scaling instead. This is not guaranteed to always work inplace (e.g. if the data is not a NumPy array or SciPy sparse CSR (Compressed Sparse Row) matrix, a copy may still be returned).
`withmean` boolean, True by default. If True, center the data before scaling. This does not work and will raise an exception when attempted on sparse matrices because centering them entails building a dense matrix which in common use cases is likely to be too large to fit in memory.

`withstd` boolean, True by default. If True, scale the data to unit variance or equivalently unit standard deviation.

Attributes

`scale` ndarray or None, shape (n_features,). Per feature relative scaling of the data. This is calculated using `np.sqrt(var)`. Equal to None when `withstd=False`.

New in version 0.17: `scale`

`mean` ndarray or None, shape (n_features,). The mean value for each feature in the training set. Equal to None when `withmean=False`.

`var` ndarray or None, shape (n_features,). The variance for each feature in the training set. Used to compute `scale`. Equal to None when `withstd=False`.

`nsamplesseen` int or array, shape (n_features,). The number of samples processed by the estimator for each feature. If there are not missing samples, the `nsamplesseen` will be an integer; otherwise, it will be an array. Will be reset on new calls to `fit` but increments across `partial_fit` calls.

See also

`scale`: Equivalent function without the estimator API (Application Programming Interface).
2274 Chapter 6: API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sklearn.decomposition.PCA Further removes the linear correlation across features with 'whiten=True'

Notes

NaNs are treated as missing values disregarded in fit and maintained in transform

We use a biased estimator for the standard deviation equivalent to numpy.std(x, ddof=0). Note that the choice of ddof is unlikely to affect model performance

For a comparison of the different scalers, transformers and normalizers see examples/preprocessing/plot_all_scaling.py

Examples

```
from sklearn.preprocessing import StandardScaler
data = 0 0 0 0 1 1 1 1
scaler = StandardScaler()
print(scaler.fit(data))
StandardScaler(copy=True, with_mean=True, with_std=True)
print(scaler.mean_)
0.5 0.5
print(scaler.transform(data))
1 1
1 1
1 1
1 1
print(scaler.transform(2 * data))
3 3
```

Methods

fit(self, X, y) Compute the mean and std to be used for later scaling

fit_transform(self, X, y) Fit to data then transform it

get_params(self, deep=True) Get parameters for this estimator

inverse_transform(self, X) Scale back the data to the original representation

partial_fit(self, X, y) Online computation of mean and std on X for later scaling

set_params(self, **params) Set the parameters of this estimator

transform(self, X) Perform standardization by centering and scaling

init(self, copy=True, with_mean=True, with_std=True)

fit(self, X, y=None)

Compute the mean and std to be used for later scaling

Parameters

X: array-like sparse matrix shape (n_samples, n_features) The data used to compute the mean and standard deviation used for later scaling along the features axis

y: ignored

634 sklearn.preprocessing: Preprocessing and Normalization 2275

scikitlearn user guide Release 0213

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

inversetransform selfXcopyNone

Scale back the data to the original representation

Parameters

Xarraylike shape nsamples nfeatures The data used to scale along the features axis

copy bool optional default None Copy the input X or not

Returns

Xtr arraylike shape nsamples nfeatures Transformed array

partialfit selfXyNone

Online computation of mean and std on X for later scaling All of X is processed as a single batch This is intended for cases when fit is not feasible due to very large number of nsamples or because X is read from a continuous stream

The algorithm for incremental mean and std is given in Equation 15ab in Chan Tony F Gene H Golub and Randall J LeVeque “Algorithms for computing the sample variance Analysis and recommendations” The American Statistician 373 1983 242247

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The data used to compute the mean and standard deviation used for later scaling along the features axis

yIgnored

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

2276 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

transform selfXcopyNone

Perform standardization by centering and scaling

Parameters

Xarraylike shape nsamples nfeatures The data used to scale along the features axis

copy bool optional default None Copy the input X or not

Examples using sklearnpreprocessingStandardScaler

- Prediction Latency
 - Classifier comparison
 - Demo of DBSCAN clustering algorithm
 - Comparing different hierarchical linkage methods on toy datasets
 - Comparing different clustering algorithms on toy datasets
 - Column Transformer with Mixed Types
 - MNIST classification using multinomial logistic L1
 - L1 Penalty and Sparsity in Logistic Regression
 - Comparing Nearest Neighbors with and without Neighborhood Components Analysis
 - Dimensionality Reduction with Neighborhood Components Analysis
 - Varying regularization in Multilayer Perceptron
 - Importance of Feature Scaling
 - Feature discretization
 - Compare the effect of different scalers on data with outliers
 - SVMAnova SVM (Support Vector Machine) with univariate feature selection
 - RBF (Radial Basis Function) SVM (Support Vector Machine) parameters
- preprocessingadddummyfeature X value Augment dataset with an additional dummy feature
- preprocessingbinarize X threshold copy Boolean thresholding of arraylike or scipysparse matrix
- preprocessinglabelbinarize y classes Binarize labels in a onevsall fashion
- preprocessingmaxabsscale X axis copy Scale each feature to the 1 1 range without breaking the sparsity
- preprocessingminmaxscale X Transforms features by scaling each feature to a given range
- preprocessingnormalize X norm axis Scale input vectors individually to unit norm vector length
- preprocessingquantiletransform X axis Transform features using quantiles information
- preprocessingrobustscale X axis Standardize a dataset along any axis
- preprocessingscale X axis withmean Standardize a dataset along any axis
- preprocessingpowertransform X method
- Power transforms are a family of parametric monotonic transformations that are applied to make data more Gaussianlike

scikitlearn user guide Release 0213

63418sklearnpreprocessing adddummyfeature

sklearnpreprocessing adddummyfeature Xvalue10

Augment dataset with an additional dummy feature

This is useful for fitting an intercept term with implementations which cannot otherwise fit it directly

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Data

value float Value to use for the dummy feature

Returns

Xarray sparse matrix shape nsamples nfeatures 1 Same data with dummy feature

added as first column

Examples

from sklearnpreprocessing import adddummyfeature

adddummyfeature0 1 1 0

array1 0 1

1 1 0

63419sklearnpreprocessing binarize

sklearnpreprocessing binarize Xthreshold00 copyTrue

Boolean thresholding of arraylike or scipysparse matrix

Read more in the User Guide

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The data to binarize element by

element scipysparse matrices should be in CSR (Compressed Sparse Row) or CSC format to avoid an unnecessary

copy

threshold float optional 00 by default Feature values below or equal to this are replaced by

0 above it by 1 Threshold may not be less than 0 for operations on sparse matrices

copy boolean optional default True set to False to perform inplace binarization and avoid a

copy if the input is already a numpy array or a scipysparse CSR (Compressed Sparse Row) CSC matrix and if axis

is 1

See also

Binarizer Performs binarization using the Transformer API (Application Programming Interface) eg as part of a preprocessing sklearn

pipelinePipeline

63420sklearnpreprocessing labelbinarize

sklearnpreprocessing labelbinarize y classes neglabel0 poslabel1

sparseoutputFalse

Binarize labels in a onevsall fashion

2278 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Several regression and binary classification algorithms are available in scikitlearn A simple way to extend these algorithms to the multiclass classification case is to use the socalled onevsall scheme This function makes it possible to compute this transformation for a fixed set of class labels known ahead of time

Parameters
yarraylike Sequence of integer labels or multilabel data to encode
classes arraylike of shape nclasses Uniquely holds the label for each class
neglabel int default 0 Value with which negative labels must be encoded
poslabel int default 1 Value with which positive labels must be encoded
sparseoutput boolean default False Set to true if output binary array is desired in CSR (Compressed Sparse Row)
sparse format

Returns
Ynumpy array or CSR (Compressed Sparse Row) matrix of shape nsamples nclasses Shape will be nsamples 1 for binary problems
See also

LabelBinarizer class used to wrap the functionality of labelbinarize and allow for fitting to classes inde pendently of the transform operation

Examples
from sklearnpreprocessing import labelbinarize
labelbinarize1 6 classes1 2 4 6
array1 0 0 0
0 0 0 1

The class ordering is preserved
labelbinarize1 6 classes1 6 4 2
array1 0 0 0
0 1 0 0

Binary targets transform to a column vector
labelbinarizeyes no no yes classesno yes
array1
0
0
1

Examples using sklearnpreprocessinglabelbinarize

- Receiver Operating Characteristic ROC
- PrecisionRecall

scikitlearn user guide Release 0213

63421sklearnpreprocessing maxabsscale

sklearnpreprocessing maxabsscale Xaxis0 copyTrue

Scale each feature to the 1 1 range without breaking the sparsity

This estimator scales each feature individually such that the maximal absolute value of each feature in the training set will be 10

This scaler can also be applied to sparse CSR (Compressed Sparse Row) or CSC matrices

Parameters

Xarraylike shape nsamples nfeatures The data

axis int 0 by default axis used to scale along If 0 independently scale each feature other wise if 1 scale each sample

copy boolean optional default is True Set to False to perform inplace scaling and avoid a copy if the input is already a numpy array

See also

MaxAbsScaler Performs scaling to the 1 1 range using the“Transformer“ API (Application Programming Interface) eg as part of a preprocessingpipelinePipeline

Notes

NaNs are treated as missing values disregarded to compute the statistics and maintained during the data transformation

For a comparison of the different scalers transformers and normalizers see examplespreprocessingplotallscalingpy

63422sklearnpreprocessing minmaxscale

sklearnpreprocessing minmaxscale Xfeaturerange0 1axis0 copyTrue

Transforms features by scaling each feature to a given range

This estimator scales and translates each feature individually such that it is in the given range on the training set ie between zero and one

The transformation is given by when axis0

$$X_{std} = \frac{X - X_{min,axis0}}{X_{max,axis0} - X_{min,axis0}}$$

Xscaled Xstd max min min

where min max featurerange

The transformation is calculated as when axis0

$$X_{scaled} = \frac{scale \cdot X - min}{X_{max,axis0} - X_{min,axis0}}$$

where scale max min Xmaxaxis0 Xminaxis0

This transformation is often used as an alternative to zero mean unit variance scaling

Read more in the User Guide

New in version 017 minmaxscale function interface to sklearnpreprocessingMinMaxScaler

Parameters

2280 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike shape nsamples nfeatures The data

featurerange tuple min max default0 1 Desired range of transformed data

axis int 0 by default axis used to scale along If 0 independently scale each feature other
wise if 1 scale each sample

copy boolean optional default is True Set to False to perform inplace scaling and avoid a
copy if the input is already a numpy array

See also

MinMaxScaler Performs scaling to a given range using the“Transformer“ API (Application Programming Interface) eg as part of a preprocess
ingsklearnpipelinePipeline

Notes

For a comparison of the different scalers transformers and normalizers see exam

plespreprocessingplotallscalingpy

Examples using sklearnpreprocessingminmaxscale

•Compare the effect of different scalers on data with outliers

63423sklearnpreprocessing normalize

sklearnpreprocessing normalize Xnorm‘l2’ axis1 copyTrue returnnormFalse

Scale input vectors individually to unit norm vector length

Read more in the User Guide

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The data to normalize element

by element scipysparse matrices should be in CSR (Compressed Sparse Row) format to avoid an unnecessary copy

norm ‘l1’ ‘l2’ or ‘max’ optional ‘l2’ by default The norm to use to normalize each non

zero sample or each nonzero feature if axis is 0

axis 0 or 1 optional 1 by default axis used to normalize the data along If 1 independently

normalize each sample otherwise if 0 normalize each feature

copy boolean optional default True set to False to perform inplace row normalization and

avoid a copy if the input is already a numpy array or a scipysparse CSR (Compressed Sparse Row) matrix and if axis
is 1

returnnorm boolean default False whether to return the computed norms

Returns

Xarraylike sparse matrix shape nsamples nfeatures Normalized input X

norms array shape nsamples if axis1 else nfeatures An array of norms along given

axis for X When X is sparse a NotImplementedError will be raised for norm ‘l1’ or ‘l2’

See also

Normalizer Performs normalization using the Transformer API (Application Programming Interface) eg as part of a preprocessing

sklearnpipelinePipeline

634sklearnpreprocessing Preprocessing and Normalization 2281

Notes

For a comparison of the different scalers transformers and normalizers see exam

plespreprocessingplotallscalingpy

63424sklearnpreprocessing quantiletransform

sklearnpreprocessing quantiletransform X axis0 nquantiles1000 out

putdistribution'uniform' ig

noreimplicitzerosFalse subsample100000

randomstateNone copy'warn'

Transform features using quantiles information

This method transforms the features to follow a uniform or a normal distribution Therefore for a given feature this transformation tends to spread out the most frequent values It also reduces the impact of marginal outliers this is therefore a robust preprocessing scheme

The transformation is applied on each feature independently First an estimate of the cumulative distribution function of a feature is used to map the original values to a uniform distribution The obtained values are then mapped to the desired output distribution using the associated quantile function Features values of newunseen data that fall below or above the fitted range will be mapped to the bounds of the output distribution Note that this transform is nonlinear It may distort linear correlations between variables measured at the same scale but renders variables measured at different scales more directly comparable

Read more in the User Guide

Parameters

Xarraylike sparse matrix The data to transform

axis int default0 Axis used to compute the means and standard deviations along If 0

transform each feature otherwise if 1 transform each sample

nquantiles int optional default1000 or nsamples Number of quantiles to be computed

It corresponds to the number of landmarks used to discretize the cumulative distribution

function If nquantiles is larger than the number of samples nquantiles is set to the

number of samples as a larger number of quantiles does not give a better approximation of

the cumulative distribution function estimator

outputdistribution str optional default'uniform' Marginal distribution for the trans

formed data The choices are 'uniform' default or 'normal'

ignoreimplicitzeros bool optional defaultFalse Only applies to sparse matrices If True

the sparse entries of the matrix are discarded to compute the quantile statistics If False

these entries are treated as zeros

subsample int optional default1e5 Maximum number of samples used to estimate the

quantiles for computational efficiency Note that the subsampling procedure may differ

for valueidentical sparse and dense matrices

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom Note that this is used by subsampling and

smoothing noise

copy boolean optional default"warn" Set to False to perform inplace transformation and

avoid a copy if the input is already a numpy array If True a copy of Xis transformed

leaving the original Xunchanged

scikitlearn user guide Release 0213

Deprecated since version 021 The default value of parameter copy will be changed from False to True in 023 The current default of False is being changed to make it more consistent with the default copy values of other functions in sklearnpreprocessing data Furthermore the current default of False may have unexpected side effects by modifying the value of Xinplace

Returns

Xtndarray or sparse matrix shape nsamples nfeatures The transformed data

See also

QuantileTransformer Performs quantilebased scaling using the Transformer API (Application Programming Interface) eg as part of a preprocessing sklearnpipelinePipeline

powertransform Maps data to a normal distribution using a power transformation

scale Performs standardization that is faster but less robust to outliers

robustscale Performs robust standardization that removes the influence of outliers but does not put out liers and inliers on the same scale

Notes

NaNs are treated as missing values disregarded in fit and maintained in transform

For a comparison of the different scalers transformers and normalizers see exam

plespreprocessingplotallscalingpy

Examples

import numpy as np

from sklearnpreprocessing import quantiletransform

rng np.random.RandomState(0)

X np.sort(rng.normal(loc=0.5, scale=0.25, size=(25, 1)), axis=0)

quantiletransform(X, nquantiles=10, random_state=0, copy=True)

array

Examples using sklearnpreprocessingquantiletransform

•Effect of transforming the targets in regression model

63425sklearnpreprocessing robustscale

sklearnpreprocessing robustscale Xaxis=0 withcentering=True withscaling=True

quantilerange(250, 750, copy=True)

Standardize a dataset along any axis

Center to the median and component wise scale according to the interquartile range

Read more in the User Guide

Parameters

Xarraylike The data to center and scale

634sklearnpreprocessing Preprocessing and Normalization 2283

scikitlearn user guide Release 0213

axis int 0 by default axis used to compute the medians and IQR along If 0 independently

scale each feature otherwise if 1 scale each sample

withcentering boolean True by default If True center the data before scaling

withscaling boolean True by default If True scale the data to unit variance or equivalently

unit standard deviation

quantilerange tuple qmin qmax 00 qmin qmax 1000 Default 250 750

1st quantile 3rd quantile IQR Quantile range used to calculate scale

New in version 018

copy boolean optional default is True set to False to perform inplace row normalization and

avoid a copy if the input is already a numpy array or a scipysparse CSR (Compressed Sparse Row) matrix and if axis

is 1

See also

RobustScaler Performs centering and scaling using the Transformer API (Application Programming Interface) eg as part of a preprocess

ingsklearnpipelinePipeline

Notes

This implementation will refuse to center scipysparse matrices since it would make them nonsparse and would

potentially crash the program with memory exhaustion problems

Instead the caller is expected to either set explicitly withcenteringFalse in that case only variance

scaling will be performed on the features of the CSR (Compressed Sparse Row) matrix or to call Xtoarray if heshe expects the

materialized dense array to fit in memory

To avoid memory copy the caller should pass a CSR (Compressed Sparse Row) matrix

For a comparison of the different scalers transformers and normalizers see exam

plespreprocessingplotallscalingpy

63426sklearnpreprocessing scale

sklearnpreprocessing scaleXaxis0 withmeanTrue withstdTrue copyTrue

Standardize a dataset along any axis

Center to the mean and component wise scale to unit variance

Read more in the User Guide

Parameters

Xarraylike sparse matrix The data to center and scale

axis int 0 by default axis used to compute the means and standard deviations along If 0

independently standardize each feature otherwise if 1 standardize each sample

withmean boolean True by default If True center the data before scaling

withstd boolean True by default If True scale the data to unit variance or equivalently unit

standard deviation

copy boolean optional default True set to False to perform inplace row normalization and

avoid a copy if the input is already a numpy array or a scipysparse CSC matrix and if axis

is 1

2284 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

See also

StandardScaler Performs scaling to unit variance using the “Transformer” API (Application Programming Interface) eg as part of a preprocessing pipeline

Notes

This implementation will refuse to center scipy sparse matrices since it would make them non sparse and would potentially crash the program with memory exhaustion problems

Instead the caller is expected to either set explicitly with mean=False in that case only variance scaling will be performed on the features of the CSC matrix or to call X.toarray if the user expects the materialized dense array to fit in memory

To avoid memory copy the caller should pass a CSC matrix

NaNs are treated as missing values disregarded to compute the statistics and maintained during the data transformation

We use a biased estimator for the standard deviation equivalent to `numpy.std(x, ddof=0)`. Note that the choice of `ddof` is unlikely to affect model performance

For a comparison of the different scalers, transformers and normalizers see examples

preprocessing

Examples using `sklearn.preprocessing.scale`

- A demo of KMeans clustering on the handwritten digits data

63427 `sklearn.preprocessing.PowerTransformer`

`sklearn.preprocessing.PowerTransformer` `X.method='warn'` `standardize=True` `copy=True`

Power transforms are a family of parametric monotonic transformations that are applied to make data more Gaussian-like. This is useful for modeling issues related to heteroscedasticity, nonconstant variance or other situations where normality is desired.

Currently `PowerTransformer` supports the Box-Cox transform and the Yeo-Johnson transform. The optimal parameter for stabilizing variance and minimizing skewness is estimated through maximum likelihood. Box-Cox requires input data to be strictly positive while Yeo-Johnson supports both positive or negative data. By default, zero mean unit variance normalization is applied to the transformed data.

Read more in the User Guide

Parameters

`X`: array-like shape (n_samples, n_features). The data to be transformed using a power transformation.

`method`: str. The power transform method. Available methods are:

- ‘yeojohnson’: 1 works with positive and negative values
- ‘boxcox’: 2 only works with strictly positive values

The default method will be changed from ‘boxcox’ to ‘yeojohnson’ in version 0.23. To suppress the FutureWarning, explicitly set the parameter.

634 `sklearn.preprocessing` Preprocessing and Normalization 2285

scikitlearn user guide Release 0213

standardize boolean defaultTrue Set to True to apply zeromean unitvariance normaliza
tion to the transformed output

copy boolean optional defaultTrue Set to False to perform inplace computation during
transformation

Returns

Xtrans arraylike shape nsamples nfeatures The transformed data

See also

PowerTransformer Equivalent transformation with the Transformer API (Application Programming Interface) eg as part of a preprocess
ingsklearnpipelinePipeline

quantiletransform Maps data to a standard normal distribution with the parameter
outputdistributionnormal

Notes

NaNs are treated as missing values disregarded in fit and maintained in transform

For a comparison of the different scalers transformers and normalizers see exam
plespreprocessingplotallscalingpy

References

12

Examples

```
import numpy as np
from sklearn.preprocessing import powertransform
data = 1 2 3 2 4 5
print(powertransform(data).methodboxcox
1332 0707
0256 0707
1076 1414
635sklearnrandomprojection Random projection
```

Random Projection transformers

Random Projections are a simple and computationally efficient way to reduce the dimensionality of the data by trading
a controlled amount of accuracy as additional variance for faster processing times and smaller model sizes

The dimensions and distribution of Random Projections matrices are controlled so as to preserve the pairwise distances
between any two samples of the dataset

The main theoretical result behind the efficiency of random projection is the JohnsonLindenstrauss lemma quoting
Wikipedia

2286 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

In mathematics the JohnsonLindenstrauss lemma is a result concerning lowdistortion embeddings of points from highdimensional into lowdimensional Euclidean space The lemma states that a small set of points in a highdimensional space can be embedded into a space of much lower dimension in such a way that distances between the points are nearly preserved The map used for the embedding is at least Lipschitz and can even be taken to be an orthogonal projection

User guide See the Random Projection section for further details

randomprojection

GaussianRandomProjection Reduce dimensionality through Gaussian random projection

randomprojection

SparseRandomProjection Reduce dimensionality through sparse random projection

6351sklearnrandomprojection GaussianRandomProjection

classssklearnrandomprojection GaussianRandomProjection ncomponents'auto'

eps01 ran

domstateNone

Reduce dimensionality through Gaussian random projection

The components of the random matrix are drawn from $N(0, 1/ncomponents)$

Read more in the User Guide

Parameters

ncomponents int or 'auto' optional default 'auto' Dimensionality of the target projection space

ncomponents can be automatically adjusted according to the number of samples in the dataset and the bound given by the JohnsonLindenstrauss lemma In that case the quality of the embedding is controlled by the eps parameter

It should be noted that JohnsonLindenstrauss lemma can yield very conservative estimated of the required number of components as it makes no assumption on the structure of the dataset

eps strictly positive float optional default01 Parameter to control the quality of the embedding according to the JohnsonLindenstrauss lemma when ncomponents is set to 'auto'

Smaller values lead to better embedding and higher number of dimensions ncomponents in the target projection space

randomstate int RandomState instance or None optional defaultNone Control the pseudo random number generator used to generate the matrix at fit time If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Attributes

ncomponent int Concrete number of components computed when ncomponents"auto"

components numpy array of shape (ncomponents, nfeatures) Random matrix used for the projection

See also

SparseRandomProjection

635sklearnrandomprojection Random projection 2287

scikitlearn user guide Release 0213

Examples

```
import numpy as np
from sklearnrandomprojection import GaussianRandomProjection
rng = np.random.RandomState(42)
X = rng.rand(100, 10000)
transformer = GaussianRandomProjection(random_state=rng)
X_new = transformer.fit_transform(X)
X_new.shape
(100, 3947)
```

Methods

```
fit(self, X, y) Generate a sparse random projection matrix
fit_transform(self, X, y) Fit to data then transform it
get_params(self, deep=True) Get parameters for this estimator
set_params(self, **params) Set the parameters of this estimator
transform(self, X) Project the data by using matrix product with the random matrix
init(self, n_components='auto', eps=0.1, random_state=None)
fit(self, X, y=None)
```

Generate a sparse random projection matrix

Parameters

X: Numpy array or scipy.sparse of shape (n_samples, n_features). Training set only the shape is used to find optimal random matrix dimensions based on the theory referenced in the afore mentioned papers

y: ignored

Returns

self

fit_transform(self, X, y=None, fit_params=None)

Fit to data then transform it

Fits transformer to X and y with optional parameters fit_params and returns a transformed version of X

Parameters

X: Numpy array of shape (n_samples, n_features). Training set

y: numpy array of shape (n_samples). Target values

Returns

X_new: numpy array of shape (n_samples, n_features_new). Transformed array

get_params(self, deep=True)

Get parameters for this estimator

Parameters

deep: boolean, optional. If True will return the parameters for this estimator and contained subobjects that are estimators

scikitlearn user guide Release 0213

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Project the data by using matrix product with the random matrix

Parameters

Xnumpy array or scipysparse of shape nsamples nfeatures The input data to project into a smaller dimensional space

Returns

Xnew numpy array or scipy sparse of shape nsamples ncomponents Projected array

6352sklearnrandomprojection SparseRandomProjection

classsklearnrandomprojection SparseRandomProjection ncomponents'auto'

density'auto' eps01

denseoutputFalse ran

domstateNone

Reduce dimensionality through sparse random projection

Sparse random matrix is an alternative to dense random projection matrix that guarantees similar embedding quality while being much more memory efficient and allowing faster computation of the projected data

If we notes 1 density the components of the random matrix are drawn from

- sqrts sqrtncomponents with probability 1 2s
- 0 with probability 1 1 s
- sqrts sqrtncomponents with probability 1 2s

Read more in the User Guide

Parameters

ncomponents int or 'auto' optional default 'auto' Dimensionality of the target projection space

ncomponents can be automatically adjusted according to the number of samples in the dataset and the bound given by the JohnsonLindenstrauss lemma In that case the quality of the embedding is controlled by the eps parameter

It should be noted that JohnsonLindenstrauss lemma can yield very conservative estimated of the required number of components as it makes no assumption on the structure of the dataset

density float in range 0 1 optional default'auto' Ratio of nonzero component in the random projection matrix

635sklearnrandomprojection Random projection 2289

scikitlearn user guide Release 0213

If density 'auto' the value is set to the minimum density as recommended by Ping Li et al 1 sqrt(n)features

Use density 1 30 if you want to reproduce the results from Achlioptas 2001

eps strictly positive float optional default0.1 Parameter to control the quality of the embedding according to the JohnsonLindenstrauss lemma when ncomponents is set to 'auto' Smaller values lead to better embedding and higher number of dimensions ncomponents in the target projection space

denseoutput boolean optional defaultFalse If True ensure that the output of the random projection is a dense numpy array even if the input and random projection matrix are both sparse In practice if the number of components is small the number of zero components in the projected data will be very small and it will be more CPU and memory efficient to use a dense representation

If False the projected data uses a sparse representation if the input is sparse randomstate int RandomState instance or None optional defaultNone Control the pseudo random number generator used to generate the matrix at fit time If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Attributes

ncomponent int Concrete number of components computed when ncomponents="auto"

components CSR (Compressed Sparse Row) matrix with shape (ncomponents, nfeatures) Random matrix used for the projection

density float in range [0, 1] Concrete density computed from when density "auto"

See also

GaussianRandomProjection

References

R0fecf191e4b81 R0fecf191e4b82

Examples

```
import numpy as np
from sklearn.random_projection import SparseRandomProjection
rng = np.random.RandomState(42)
X = rng.rand(100, 10000)
transformer = SparseRandomProjection(random_state=rng)
X_new = transformer.fit_transform(X)
X_new.shape
(100, 3947)
# very few components are nonzero
np.mean(transformer.components_ > 0)
0.0100
```


scikitlearn user guide Release 0213

Methods

fitself X y Generate a sparse random projection matrix
fittransform self X y Fit to data then transform it
getparams self deep Get parameters for this estimator
setparams self params Set the parameters of this estimator
transform self X Project the data by using matrix product with the random matrix
init selfncomponents'auto' density'auto' eps01 denseoutputFalse random
domstateNone
fitselfXyNone

Generate a sparse random projection matrix

Parameters

Xnumpy array or scipy sparse of shape nsamples nfeatures Training set only the shape is used to find optimal random matrix dimensions based on the theory referenced in the afore mentioned papers

yIgnored

Returns

self

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

635sklearnrandomprojection Random projection 2291

scikitlearn user guide Release 0213

transform selfX

Project the data by using matrix product with the random matrix

Parameters

Xnumpy array or scipysparse of shape nsamples nfeatures The input data to project into a smaller dimensional space

Returns

Xnew numpy array or scipy sparse of shape nsamples ncomponents Projected array

Examples using sklearnrandomprojectionSparseRandomProjection

- The JohnsonLindenstrauss bound for embedding with random projections
- Manifold learning on handwritten digits Locally Linear Embedding Isomap

randomprojection

johnsonlindenstraussmindim Find a ‘safe’ number of components to randomly project to

6353sklearnrandomprojection johnsonlindenstraussmindim

sklearnrandomprojection johnsonlindenstraussmindim nsamples eps01

Find a ‘safe’ number of components to randomly project to

The distortion introduced by a random projection ponly changes the distance between two points by a factor 1
eps in an euclidean space with good probability The projection pis an epseembedding as defined by

1 eps u v2 pu pv2 1 eps u v2

Where u and v are any rows taken from a dataset of shape nsamples nfeatures eps is in 0 1 and p is a
projection by a random Gaussian N0 1 matrix with shape ncomponents nfeatures or a sparse Achlioptas
matrix

The minimum number of components to guarantee the epseembedding is given by

ncomponents 4 lognsamples eps2 2 eps3 3

Note that the number of dimensions is independent of the original number of features but instead depends on
the size of the dataset the larger the dataset the higher is the minimal dimensionality of an epseembedding

Read more in the User Guide

Parameters

nsamples int or numpy array of int greater than 0 Number of samples If an array is given
it will compute a safe number of components arraywise

eps float or numpy array of float in 01 optional default01 Maximum distortion rate as
defined by the JohnsonLindenstrauss lemma If an array is given it will compute a safe
number of components arraywise

Returns

ncomponents int or numpy array of int The minimal number of components to guarantee
with good probability an epseembedding with nsamples

2292 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

References

12

Examples

johnsonlindenstraussmindim1e6 eps05

663

johnsonlindenstraussmindim1e6 eps05 01 001

array 663 11841 1112658

johnsonlindenstraussmindim1e4 1e5 1e6 eps01

array 7894 9868 11841

Examples using sklearnrandomprojectionjohnsonlindenstraussmindim

•The JohnsonLindenstrauss bound for embedding with random projections

636sklearnsemisupervised SemiSupervised Learning

Thesklearnsemisupervised module implements semisupervised learning algorithms These algorithms

utilized small amounts of labeled data and large amounts of unlabeled data for classification tasks This module

includes Label Propagation

User guide See the SemiSupervised section for further details

semisupervisedLabelPropagation kernel

Label Propagation classifier

semisupervisedLabelSpreading kernel

LabelSpreading model for semisupervised learning

6361sklearnsemisupervised LabelPropagation

classsklearnsemisupervised LabelPropagation kernel'rbf' gamma20 nneighbors7

maxiter1000 tol0001 njobsNone

Label Propagation classifier

Read more in the User Guide

Parameters

kernel 'knn' 'rbf' callable String identifier for kernel function to use or the kernel function

itself Only 'rbf' and 'knn' strings are valid inputs The function passed should take two

inputs each of shape nsamples nfeatures and return a nsamples nsamples shaped

weight matrix

gamma float Parameter for rbf kernel

nneighbors integer 0 Parameter for knn kernel

636sklearnsemisupervised SemiSupervised Learning 2293

scikitlearn user guide Release 0213
maxiter integer Change maximum number of iterations allowed
tolfloat Convergence tolerance threshold to consider the system at steady state
njobs int or None optional defaultNone The number of parallel jobs to run None means
1 unless in a joblibparallelbackend context1means using all processors See
Glossary for more details

Attributes
Xarray shape nsamples nfeatures Input array
classes array shape nclasses The distinct labels used in classifying instances
labeldistributions array shape nsamples nclasses Categorical distribution for each
item
transduction array shape nsamples Label assigned to each item via the transduction
niter int Number of iterations run

See also
LabelSpreading Alternate label propagation strategy more robust to noise
References
Xiaojin Zhu and Zoubin Ghahramani Learning from labeled and unlabeled data with label propagation Tech
nical Report CMUCALD02107 Carnegie Mellon University 2002 <http://pages.cse.cmu.edu/jerryzhupub>
CMUCALD02107pdf

Examples
import numpy as np
from sklearn import datasets
from sklearnsemisupervised import LabelPropagation
labelpropmodel LabelPropagation
iris datasetsloadiris
rng np.random.RandomState(42)
randomunlabeledpoints rng.random(len(iris.target) * 0.3)
labels np.copy(iris.target)
labels[randomunlabeledpoints] = 1
labelpropmodel.fit(iris.data, labels)

LabelPropagation
Methods
fitself X y
getparams self deep Get parameters for this estimator
predict self X Performs inductive inference across the model
predictproba self X Predict probability for each possible outcome
score self X y sampleweight Returns the mean accuracy on the given test data and
labels

Continued on next page
2294 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Table 6270 – continued from previous page

setparams self params Set the parameters of this estimator

init selfkernel'rbf' gamma20 nneighbors7 maxiter1000 tol0001 njobsNone

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Performs inductive inference across the model

Parameters

Xarraylike shape nsamples nfeatures

Returns

yarraylike shape nsamples Predictions for input data

predictproba selfX

Predict probability for each possible outcome

Compute the probability estimates for each single sample in X and each possible outcome seen during

training categorical distribution

Parameters

Xarraylike shape nsamples nfeatures

Returns

probabilities array shape nsamples nclasses Normalized probability distributions

across class labels

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each

sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

636sklearnsemisupervised SemiSupervised Learning 2295

scikitlearn user guide Release 0213

Returns

self

6362sklearnsemisupervised LabelSpreading

classsklearnsemisupervised LabelSpreading kernel'rbf' gamma20 nneighbors7

alpha02 maxiter30 tol0001

njobsNone

LabelSpreading model for semisupervised learning

This model is similar to the basic Label Propagation algorithm but uses affinity matrix based on the normalized graph Laplacian and soft clamping across the labels

Read more in the User Guide

Parameters

kernel 'knn' 'rbf' callable String identifier for kernel function to use or the kernel function

itself Only 'rbf' and 'knn' strings are valid inputs The function passed should take two

inputs each of shape nsamples nfeatures and return a nsamples nsamples shaped

weight matrix

gamma float parameter for rbf kernel

nneighbors integer 0 parameter for knn kernel

alpha float Clamping factor A value in 0 1 that specifies the relative amount that an instance

should adopt the information from its neighbors as opposed to its initial label alpha0

means keeping the initial label information alpha1 means replacing all initial information

maxiter integer maximum number of iterations allowed

tolfloat Convergence tolerance threshold to consider the system at steady state

njobs int or None optional defaultNone The number of parallel jobs to run None means

1 unless in a joblibparallelbackend context1means using all processors See

Glossary for more details

Attributes

Xarray shape nsamples nfeatures Input array

classes array shape nclasses The distinct labels used in classifying instances

labeldistributions array shape nsamples nclasses Categorical distribution for each

item

transduction array shape nsamples Label assigned to each item via the transduction

niter int Number of iterations run

See also

LabelPropagation Unregularized graph based semisupervised learning

References

Dengyong Zhou Olivier Bousquet Thomas Navin Lal Jason Weston Bernhard Schoelkopf Learning with

local and global consistency 2004 <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.115.3219>

2296 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Examples

```
import numpy as np
from sklearn import datasets
from sklearn.semisupervised import LabelSpreading
labelpropmodel = LabelSpreading
iris = datasets.load_iris
rng = np.random.RandomState(42)
random_unlabeled_points = rng.rand(len(iris.target) * 0.3)
labels = np.copy(iris.target)
labels[random_unlabeled_points] = 1
labelpropmodel.fit(iris.data, labels)
```

LabelSpreading

Methods

`fit(self, X, y)` Fit a semisupervised label propagation model based on `X` and `y`.

`get_params(self, deep=True)` Get parameters for this estimator.

`predict(self, X)` Performs inductive inference across the model.

`predict_proba(self, X)` Predict probability for each possible outcome.

`score(self, X, y, sample_weight)` Returns the mean accuracy on the given test data and labels.

`set_params(self, **params)` Set the parameters of this estimator.

`__init__(self, kernel='rbf', gamma=0.2, n_neighbors=7, alpha=0.2, max_iter=30, tol=0.001, n_jobs=None)`

`fit(self, X, y)` Fit a semisupervised label propagation model based on `X` and `y`. All the input data is provided as matrix `X` (labeled and unlabeled) and corresponding label matrix `y` with a dedicated marker value for unlabeled samples.

Parameters

- `X`: array-like shape `(n_samples, n_features)`. A `n_samples` by `n_features` matrix will be created from this.
- `y`: array-like shape `(n_samples,)`. Labeled samples and unlabeled points are marked as 1. All unlabeled samples will be transductively assigned labels.

Returns

`self`: returns an instance of `LabelSpreading`.

`get_params(self, deep=True)` Get parameters for this estimator.

Parameters

- `deep`: boolean, optional. If `True` will return the parameters for this estimator and contained subobjects that are estimators.

Returns

`params`: mapping of string to any Parameter names mapped to their values.

scikitlearn user guide Release 0213

predictselfX

Performs inductive inference across the model

Parameters

Xarraylike shape nsamples nfeatures

Returns

yarraylike shape nsamples Predictions for input data

predictproba selfX

Predict probability for each possible outcome

Compute the probability estimates for each single sample in X and each possible outcome seen during training categorical distribution

Parameters

Xarraylike shape nsamples nfeatures

Returns

probabilities array shape nsamples nclasses Normalized probability distributions

across class labels

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnsemisupervisedLabelSpreading

- Decision boundary of label propagation versus SVM (Support Vector Machine) on the Iris dataset
- Label Propagation learning a complex structure
- Label Propagation digits Demonstrating performance
- Label Propagation digits active learning

2298 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

637sklearnsvm Support Vector Machines

The sklearnsvm module includes Support Vector Machine algorithms

User guide See the Support Vector Machines section for further details

6371 Estimators

svmLinearSVC penalty loss dual tol C Linear Support Vector Classification

svmLinearSVR epsilon tol C loss Linear Support Vector Regression

svmNuSVC nu kernel degree gamma NuSupport Vector Classification

svmNuSVR nu C kernel degree gamma Nu Support Vector Regression

svmOneClassSVM kernel degree gamma Unsupervised Outlier Detection

svmSVC C kernel degree gamma coef0 CSupport Vector Classification

svmSVR kernel degree gamma coef0 tol EpsilonSupport Vector Regression

sklearnsvm LinearSVC

class sklearnsvm LinearSVC penalty' l2' loss'squaredhinge' dualTrue tol00001

C10 multiclass'ovr' fitinterceptTrue interceptscaling1

class weightNone verbose0 randomstateNone maxiter1000

Linear Support Vector Classification

Similar to SVC (Support Vector Classification) with parameter kernel'linear' but implemented in terms of liblinear rather than libsvm so it has more flexibility in the choice of penalties and loss functions and should scale better to large numbers of samples

This class supports both dense and sparse input and the multiclass support is handled according to a one-vs-the-rest scheme

Read more in the User Guide

Parameters

penalty string 'l1' or 'l2' default'l2' Specifies the norm used in the penalization The 'l2' penalty is the standard used in SVC (Support Vector Classification) The 'l1' leads to coef vectors that are sparse

loss string 'hinge' or 'squaredhinge' default'squaredhinge' Specifies the loss function 'hinge' is the standard SVM (Support Vector Machine) loss used eg by the SVC (Support Vector Classification) class while 'squaredhinge' is the square of the hinge loss

dual bool defaultTrue Select the algorithm to either solve the dual or primal optimization problem Prefer dualFalse when n samples n features

tol float optional default1e4 Tolerance for stopping criteria

C float optional default10 Penalty parameter C of the error term

multiclass string 'ovr' or 'crammersinger' default'ovr' Determines the multiclass strategy if y contains more than two classes ovr trains n classes one-vs-rest classifiers while crammersinger optimizes a joint objective over all classes While crammersinger is interesting from a theoretical perspective as it is consistent it is seldom used in practice as it rarely leads to better accuracy and is more expensive to compute If crammersinger is chosen the options loss penalty and dual will be ignored

fitintercept boolean optional defaultTrue Whether to calculate the intercept for this model If set to false no intercept will be used in calculations ie data is expected to be centered

637sklearnsvm Support Vector Machines 2299

scikitlearn user guide Release 0213

be already centered

interceptscaling float optional default1 When selffitintercept is True instance vector

x becomesx selfinterceptscaling ie a “synthetic” feature with constant

value equals to interceptscaling is appended to the instance vector The intercept becomes

interceptscaling synthetic feature weight Note the synthetic feature weight is subject to

l1l2 regularization as all other features To lessen the effect of regularization on synthetic

feature weight and therefore on the intercept interceptscaling has to be increased

classweight dict ‘balanced’ optional Set the parameter C of class i to

classweighti Cfor SVC (Support Vector Classification) If not given all classes are supposed to have weight

one The “balanced” mode uses the values of y to automatically adjust weights inversely

proportional to class frequencies in the input data as nsamples nclasses

npbincounty

verbose int default0 Enable verbose output Note that this setting takes advantage of a per
process runtime setting in liblinear that if enabled may not work properly in a multithreaded
context

randomstate int RandomState instance or None optional defaultNone The seed of the

pseudo random number generator to use when shuffling the data for the dual coordi

nate descent if dualTrue WhendualFalse the underlying implementation of

LinearSVC is not random and randomstate has no effect on the results If int ran

domstate is the seed used by the random number generator If RandomState instance ran

domstate is the random number generator If None the random number generator is the

RandomState instance used by nprandom

maxiter int default1000 The maximum number of iterations to be run

Attributes

coef array shape nfeatures if nclasses 2 else nclasses nfeatures Weights as

signed to the features coefficients in the primal problem This is only available in the case
of a linear kernel

coef is a readonly property derived from rawcoef that follows the internal memory

layout of liblinear

intercept array shape 1 if nclasses 2 else nclasses Constants in decision function

See also

SVC (Support Vector Classification) Implementation of Support Vector Machine classifier using libsvm the kernel can be nonlinear but its

SMO algorithm does not scale to large number of samples as LinearSVC does Furthermore SVC (Support Vector Classification) multi

class mode is implemented using one vs one scheme while LinearSVC uses one vs the rest It is possible to

implement one vs the rest with SVC (Support Vector Classification) by using the sklearnmulticlassOneVsRestClassifier

wrapper Finally SVC (Support Vector Classification) can fit dense data without memory copy if the input is Ccontiguous Sparse data

will still incur memory copy though

sklearnlinearmodelSGDClassifier SGDClassifier can optimize the same cost function as Lin

earSVC by adjusting the penalty and loss parameters In addition it requires less memory allows incre

mental online learning and implements various loss functions and regularization regimes

Notes

The underlying C implementation uses a random number generator to select features when fitting the model

It is thus not uncommon to have slightly different results for the same input data If that happens try with a

smallertol parameter

2300 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The underlying implementation liblinear uses a sparse internal representation for the data that will incur a memory copy

Predict output may not match that of standalone liblinear in certain cases See differences from liblinear in the narrative documentation

References

LIBLINEAR A Library for Large Linear Classification

Examples

```
from sklearnsvm import LinearSVC
from sklearnndatasets import makeclassification
X y makeclassificationnfeatures4 randomstate0
clf LinearSVCrandomstate0 tol1e5
clffitX y
LinearSVCC10 classweightNone dualTrue fitinterceptTrue
interceptscaling1 losssquaredhinge maxiter1000
multiclassovr penaltyl2 randomstate0 tol1e05 verbose0
printclfcoef
0085 0394 0498 0375
printclfintercept
0284
printclfpredict0 0 0 0
1
```

Methods

- decisionfunction self X Predict confidence scores for samples
- densify self Convert coefficient matrix to dense array format
- fitself X y sampleweight Fit the model according to the given training data
- getparams self deep Get parameters for this estimator
- predict self X Predict class labels for samples in X
- score self X y sampleweight Returns the mean accuracy on the given test data and labels
- setparams self params Set the parameters of this estimator
- sparsify self Convert coefficient matrix to sparse format
- init selfpenalty'l2' loss'squaredhinge' dualTrue tol00001 C10 multiclass'ovr' fitinterceptTrue interceptscaling1 classweightNone verbose0 ran
- domstateNone maxiter1000
- decisionfunction selfX
- Predict confidence scores for samples
- The confidence score for a sample is the signed distance of that sample to the hyperplane

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

637sklearnsvm Support Vector Machines 2301

scikitlearn user guide Release 0213

array shapensamples if nclasses > 2 else nsamples nclasses Confidence scores per sample class combination In the binary case confidence score for selfclasses1 where 0 means this class would be predicted

densifyself

Convert coefficient matrix to dense array format

Converts the coef member back to a numpyndarray This is the default format of coef and is required for fitting so calling this method is only required on models that have previously been sparsified otherwise it is a noop

Returns

self estimator

fitselfXysampleweightNone

Fit the model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vector where nsamples is the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target vector relative to X

sampleweight arraylike shape nsamples optional Array of weights that are assigned to individual samples If not provided then each sample is given unit weight

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict class labels for samples in X

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Predicted class label per sample

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

2302 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

sparsify self

Convert coefficient matrix to sparse format

Converts the coef member to a scipysparse matrix which for L1regularized models can be much more

memory and storageefficient than the usual numpyndarray representation

Theintercept member is not converted

Returns

self estimator

Notes

For nonsparse models ie when there are not many zeros in coef this may actually increase memory

usage so use this method with care A rule of thumb is that the number of zero elements which can be

computed with coef 0sum must be more than 50 for this to provide significant benefits

After calling this method further fitting with the partialfit method if any will not work until you call

densify

Examples using sklearnsvmLinearSVC

- Explicit feature map approximation for RBF (Radial Basis Function) kernels
- Comparison of Calibration of Classifiers
- Probability Calibration curves
- Selecting dimensionality reduction with Pipeline and GridSearchCV
- Column Transformer with Heterogeneous Data Sources
- Pipeline Anova SVM (Support Vector Machine)
- Balance model complexity and crossvalidated score
- PrecisionRecall
- Feature discretization
- Plot different SVM (Support Vector Machine) classifiers in the iris dataset
- Scaling the regularization parameter for SVCs
- Classification of text documents using sparse features

637sklearnsvm Support Vector Machines 2303

scikitlearn user guide Release 0213

sklearnsvm LinearSVR

classsklearnsvm LinearSVR epsilon00 tol00001 C10 loss'epsiloninsensitive'

fitinterceptTrue interceptscaling10 dualTrue verbose0

randomstateNone maxiter1000

Linear Support Vector Regression

Similar to SVR (Support Vector Regression) with parameter kernel'linear' but implemented in terms of liblinear rather than libsvm so it has more flexibility in the choice of penalties and loss functions and should scale better to large numbers of samples

This class supports both dense and sparse input

Read more in the User Guide

Parameters

epsilon float optional default00 Epsilon parameter in the epsiloninsensitive loss function

Note that the value of this parameter depends on the scale of the target variable y If unsure

setepsilon0

tolfloat optional default1e4 Tolerance for stopping criteria

Cfloat optional default10 Penalty parameter C of the error term The penalty is a squared

L2 penalty The bigger this parameter the less regularization is used

loss string optional default'epsiloninsensitive' Specifies the loss function The epsilon

insensitive loss standard SVR (Support Vector Regression) is the L1 loss while the squared epsiloninsensitive loss

'squaredepsiloninsensitive' is the L2 loss

fitintercept boolean optional defaultTrue Whether to calculate the intercept for this

model If set to false no intercept will be used in calculations ie data is expected to be already centered

interceptscaling float optional default1 When selffitintercept is True instance vec

tor x becomes x selfinterceptscaling ie a "synthetic" feature with constant value

equals to interceptscaling is appended to the instance vector The intercept becomes in

terceptscaling synthetic feature weight Note the synthetic feature weight is subject to

L1L2 regularization as all other features To lessen the effect of regularization on synthetic

feature weight and therefore on the intercept interceptscaling has to be increased

dual bool defaultTrue Select the algorithm to either solve the dual or primal optimization

problem Prefer dualFalse when nsamples nfeatures

verbose int default0 Enable verbose output Note that this setting takes advantage of a per process runtime setting in liblinear that if enabled may not work properly in a multithreaded context

randomstate int RandomState instance or None optional defaultNone The seed of the

pseudo random number generator to use when shuffling the data If int randomstate is

the seed used by the random number generator If RandomState instance randomstate is

the random number generator If None the random number generator is the RandomState

instance used by nprandom

maxiter int default1000 The maximum number of iterations to be run

Attributes

coef array shape nfeatures if nclasses 2 else nclasses nfeatures Weights as

signed to the features coefficients in the primal problem This is only available in the case of a linear kernel

2304 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

coef is a readonly property derived from rawcoef that follows the internal memory layout of liblinear

intercept array shape 1 if nclasses 2 else nclasses Constants in decision function

See also

LinearSVC Implementation of Support Vector Machine classifier using the same library as this class liblin

ear

SVR (Support Vector Regression) Implementation of Support Vector Machine regression using libsvm the kernel can be nonlinear but its SMO algorithm does not scale to large number of samples as LinearSVC does

sklearnlinearmodelSGDRegressor SGDRegressor can optimize the same cost function as Lin

earSVR by adjusting the penalty and loss parameters In addition it requires less memory allows incre

mental online learning and implements various loss functions and regularization regimes

Examples

```
from sklearnsvm import LinearSVR
```

```
from sklearndatasets import makeregression
```

```
X y makeregressionnfeatures4 randomstate0
```

```
regr LinearSVRrandomstate0 tol1e5
```

```
regrfitX y
```

```
LinearSVRC10 dualTrue epsilon00 fitinterceptTrue
```

```
interceptscaling10 lossepsiloninsensitive maxiter1000
```

```
randomstate0 tol1e05 verbose0
```

```
printregrcoef
```

```
1635 2691 4230 6047
```

```
printregrintercept
```

```
429
```

```
printregrpredict0 0 0 0
```

```
429
```

Methods

fitself X y sampleweight Fit the model according to the given training data

getparams self deep Get parameters for this estimator

predict self X Predict using the linear model

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfepsilon00 tol00001 C10 loss'epsiloninsensitive' fitinterceptTrue interceptscaling10 dualTrue verbose0 randomstateNone maxiter1000

fitselfXysampleweightNone

Fit the model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vector where

nsamples in the number of samples and nfeatures is the number of features

yarraylike shape nsamples Target vector relative to X

637sklearnsvm Support Vector Machines 2305

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples optional Array of weights that are as signed to individual samples If not provided then each sample is given unit weight

Returns

self object

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Predict using the linear model

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Samples

Returns

Carray shape nsamples Returns predicted values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

2306 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

sklearnsvm NuSVC

classsklearnsvm NuSVCnu05 kernel'rbf' degree3 gamma'autodeprecated'

coef000 shrinkingTrue probabilityFalse tol0001

cachesize200 classweightNone verboseFalse maxiter1 deci

sionfunctionshape'ovr' randomstateNone

NuSupport Vector Classification

Similar to SVC (Support Vector Classification) but uses a parameter to control the number of support vectors

The implementation is based on libsvm

Read more in the User Guide

Parameters

nu float optional default05 An upper bound on the fraction of training errors and a lower

bound of the fraction of support vectors Should be in the interval 0 1

kernel string optional default'rbf' Specifies the kernel type to be used in the algorithm

It must be one of 'linear' 'poly' 'rbf' 'sigmoid' 'precomputed' or a callable If none is

given 'rbf' will be used If a callable is given it is used to precompute the kernel matrix

degree int optional default3 Degree of the polynomial kernel function 'poly' Ignored

by all other kernels

gamma float optional default'auto' Kernel coefficient for 'rbf' 'poly' and 'sigmoid'

Current default is 'auto' which uses 1 / nfeatures if gammascale is passed then it

uses 1 / nfeatures Xvar as value of gamma The current default of gamma 'auto'

will change to 'scale' in version 022 'autodeprecated' a deprecated version of 'auto' is

used as a default indicating that no explicit value of gamma was passed

coef0 float optional default00 Independent term in kernel function It is only significant

in 'poly' and 'sigmoid'

shrinking boolean optional defaultTrue Whether to use the shrinking heuristic

probability boolean optional defaultFalse Whether to enable probability estimates This

must be enabled prior to calling fit and will slow down that method

tol float optional default1e3 Tolerance for stopping criterion

cachesize float optional Specify the size of the kernel cache in MB

classweight dict 'balanced' optional Set the parameter C of class i to classweightiC

for SVC (Support Vector Classification) If not given all classes are supposed to have weight one The "balanced" mode

uses the values of y to automatically adjust weights inversely proportional to class frequen

cies asnsamples nclasses npbincounty

637sklearnsvm Support Vector Machines 2307

scikitlearn user guide Release 0213

verbose bool default False Enable verbose output Note that this setting takes advantage of a perprocess runtime setting in libsvm that if enabled may not work properly in a multithreaded context

maxiter int optional default1 Hard limit on iterations within solver or 1 for no limit

decisionfunctionshape 'ovo' 'ovr' default'ovr' Whether to return a onevsrest 'ovr'

decision function of shape nsamples nclasses as all other classifiers or the original onevsone 'ovo' decision function of libsvm which has shape nsamples nclasses

nclasses 1 2

Changed in version 019 decisionfunctionshape is 'ovr' by default

New in version 017 decisionfunctionshape'ovr' is recommended

Changed in version 017 Deprecated decisionfunctionshape'ovo' and None

randomstate int RandomState instance or None optional defaultNone The seed of the pseudo random number generator used when shuffling the data for probability estimates

If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Attributes

support arraylike shape nSV Indices of support vectors

supportvectors arraylike shape nSV nfeatures Support vectors

nsupport arraylike dtypeint32 shape nclass Number of support vectors for each

class

dualcoef array shape nclass1 nSV Coefficients of the support vector in the decision

function For multiclass coefficient for all 1vs1 classifiers The layout of the coefficients in the multiclass case is somewhat nontrivial See the section about multiclass classification in the SVM (Support Vector Machine) section of the User Guide for details

coef array shape nclass nclass1 2 nfeatures Weights assigned to the features

coefficients in the primal problem This is only available in the case of a linear kernel

coef is readonly property derived from dualcoef andsupportvectors

intercept array shape nclass nclass1 2 Constants in decision function

See also

SVC (Support Vector Classification) Support Vector Machine for classification using libsvm

LinearSVC Scalable linear Support Vector Machine for classification using liblinear

Notes

References LIBSVM A Library for Support Vector Machines

Examples

```
import numpy as np
```

```
X = np.array([1 2 1 1 1 2 1
```

```
y = np.array([1 1 2 2
```

```
from sklearnsvm import NuSVC
```

2308 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

clf NuSVCgamma scale

clffitX y

NuSVCcachesize200 classweightNone coef000

decisionfunctionshapeovr degree3 gamma scale kernelrbf

maxiter1 nu05 probabilityFalse randomstateNone

shrinkingTrue tol0001 verboseFalse

printclfpredict08 1

1

Methods

decisionfunction self X Evaluates the decision function for the samples in X

fitself X y sampleweight Fit the SVM (Support Vector Machine) model according to the given training data

getparams self deep Get parameters for this estimator

predict self X Perform classification on samples in X

score self X y sampleweight Returns the mean accuracy on the given test data and

labels

setparams self params Set the parameters of this estimator

init selfnu05 kernel'rbf' degree3 gamma'autodeprecated' coef000 shrink

ingTrue probabilityFalse tol0001 cachesize200 classweightNone ver

boseFalse maxiter1 decisionfunctionshape'ovr' randomstateNone

decisionfunction selfX

Evaluates the decision function for the samples in X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Xarraylike shape nsamples nclasses nclasses1 2 Returns the decision func

tion of the sample for each class in the model If decisionfunctionshape'ovr' the shape

is nsamples nclasses

Notes

If decisionfunctionshape'ovo' the function values are proportional to the distance of the samples X to

the separating hyperplane If the exact distances are required divide the function values by the norm of

the weight vector coef See also this question for further details If decisionfunctionshape'ovr'

the decision function is a monotonic transformation of ovo decision function

fitselfXysampleweightNone

Fit the SVM (Support Vector Machine) model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where

nsamples is the number of samples and nfeatures is the number of features For ker

nel"precomputed" the expected shape of X is nsamples nsamples

yarraylike shape nsamples Target values class labels in classification real numbers

in regression

637sklearnsvm Support Vector Machines 2309

scikitlearn user guide Release 0213

sampleweight arraylike shape nsamples Persample weights Rescale C per sample
Higher weights force the classifier to put more emphasis on these points

Returns
self object

Notes
If X and y are not Cordered and contiguous arrays of npfloat64 and X is not a scipysparsecsrmatrix X
andor y may be copied
If X is a dense array then the other methods will not support sparse matrices as input

getparams selfdeepTrue
Get parameters for this estimator

Parameters
deep boolean optional If True will return the parameters for this estimator and contained
subobjects that are estimators

Returns
params mapping of string to any Parameter names mapped to their values

predictselfX
Perform classification on samples in X
For an oneclass model 1 or 1 is returned

Parameters
Xarraylike sparse matrix shape nsamples nfeatures For kernel"precomputed"
the expected shape of X is nsamplestest nsamplestrain

Returns
ypred array shape nsamples Class labels for samples in X

predictlogproba
Compute log probabilities of possible outcomes for samples in X
The model need to have probability information computed at training time fit with attribute
probability set to True

Parameters
Xarraylike shape nsamples nfeatures For kernel"precomputed" the expected
shape of X is nsamplestest nsamplestrain

Returns
Tarraylike shape nsamples nclasses Returns the logprobabilities of the sample for
each class in the model The columns correspond to the classes in sorted order as they
appear in the attribute classes

Notes
The probability model is created using cross validation so the results can be slightly different than those
obtained by predict Also it will produce meaningless results on very small datasets

2310 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

predictproba

Compute probabilities of possible outcomes for samples in X

The model need to have probability information computed at training time fit with attribute

probability set to True

Parameters

Xarraylike shape nsamples nfeatures For kernel”precomputed” the expected

shape of X is nsamplestest nsamplestrain

Returns

Tarraylike shape nsamples nclasses Returns the probability of the sample for each

class in the model The columns correspond to the classes in sorted order as they appear

in the attribute classes

Notes

The probability model is created using cross validation so the results can be slightly different than those

obtained by predict Also it will produce meaningless results on very small datasets

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each

sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it’s possible to update each component

of a nested object

Returns

self

Examples using sklearnsvmNuSVC

•Nonlinear SVM (Support Vector Machine)

637sklearnsvm Support Vector Machines 2311

scikitlearn user guide Release 0213

sklearnsvm NuSVR

classsklearnsvm NuSVRnu05 C10 kernel'rbf' degree3 gamma'autodeprecated'

coef000 shrinkingTrue tol0001 cachesize200 verboseFalse

maxiter1

Nu Support Vector Regression

Similar to NuSVC for regression uses a parameter nu to control the number of support vectors However unlike NuSVC where nu replaces C here nu replaces the parameter epsilon of epsilonSVR

The implementation is based on libsvm

Read more in the User Guide

Parameters

nu float optional An upper bound on the fraction of training errors and a lower bound of the fraction of support vectors Should be in the interval 0 1 By default 05 will be taken

C float optional default10 Penalty parameter C of the error term

kernel string optional default'rbf' Specifies the kernel type to be used in the algorithm

It must be one of 'linear' 'poly' 'rbf' 'sigmoid' 'precomputed' or a callable If none is

given 'rbf' will be used If a callable is given it is used to precompute the kernel matrix

degree int optional default3 Degree of the polynomial kernel function 'poly' Ignored by all other kernels

gamma float optional default'auto' Kernel coefficient for 'rbf' 'poly' and 'sigmoid'

Current default is 'auto' which uses 1 / nfeatures if gammascale is passed then it

uses 1 / nfeatures Xvar as value of gamma The current default of gamma 'auto'

will change to 'scale' in version 022 'autodeprecated' a deprecated version of 'auto' is

used as a default indicating that no explicit value of gamma was passed

coef0 float optional default00 Independent term in kernel function It is only significant in 'poly' and 'sigmoid'

shrinking boolean optional defaultTrue Whether to use the shrinking heuristic

tol float optional default1e3 Tolerance for stopping criterion

cachesize float optional Specify the size of the kernel cache in MB

verbose bool default False Enable verbose output Note that this setting takes advantage

of a perprocess runtime setting in libsvm that if enabled may not work properly in a

multithreaded context

maxiter int optional default1 Hard limit on iterations within solver or 1 for no limit

Attributes

support arraylike shape nSV Indices of support vectors

supportvectors arraylike shape nSV nfeatures Support vectors

dualcoef array shape 1 nSV Coefficients of the support vector in the decision function

coef array shape 1 nfeatures Weights assigned to the features coefficients in the primal problem This is only available in the case of a linear kernel

coef is readonly property derived from dualcoef andsupportvectors

intercept array shape 1 Constants in decision function

2312 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

See also

NuSVC Support Vector Machine for classification implemented with libsvm with a parameter to control the number of support vectors

SVR (Support Vector Regression) epsilon Support Vector Machine for regression implemented with libsvm

Notes

References

LIBSVM A Library for Support Vector Machines

Examples

```
from sklearnsvm import NuSVR
import numpy as np
nsamples nfeatures 10 5
nprandomseed0
y nprandomrandnnsamples
X nprandomrandnnsamples nfeatures
clf NuSVRgamma C10 nu01
clffitX y
NuSVRC10 cachesize200 coef000 degree3 gamma C10 nu01
kernelrbf maxiter1 nu01 shrinkingTrue tol0001
verboseFalse
```

Methods

fitself X y sampleweight Fit the SVM (Support Vector Machine) model according to the given training data

getparams self deep Get parameters for this estimator

predict self X Perform regression on samples in X

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfnu05 C10 kernel'rbf' degree3 gamma'autodeprecated' coef000 shrink

ingTrue tol0001 cachesize200 verboseFalse maxiter1

fitselfXysampleweightNone

Fit the SVM (Support Vector Machine) model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where nsamples is the number of samples and nfeatures is the number of features For kernel"precomputed" the expected shape of X is nsamples nsamples

yarraylike shape nsamples Target values class labels in classification real numbers in regression

sampleweight arraylike shape nsamples Persample weights Rescale C per sample

Higher weights force the classifier to put more emphasis on these points

Returns

637sklearnsvm Support Vector Machines 2313

scikitlearn user guide Release 0213

self object

Notes

If X and y are not Cordered and contiguous arrays of npfloat64 and X is not a scipysparsecsrmatrix X andor y may be copied

If X is a dense array then the other methods will not support sparse matrices as input

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Perform regression on samples in X

For an oneclass model 1 inlier or 1 outlier is returned

Parameters

Xarraylike sparse matrix shape nsamples nfeatures For kernel”precomputed”

the expected shape of X is nsamplestest nsamplestrain

Returns

ypred array shape nsamples

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

2314 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

To specify the default value manually and avoid the warning please either call `metricsr2score` directly or make a custom scorer with `metricsmakescorer` the builtin scorer `r2` uses

`multioutputuniformaverage`

`setparams selfparams`

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form `componentparameter` so that it's possible to update each component

of a nested object

Returns

`self`

Examples using `sklearnsvmNuSVR`

•Model Complexity Influence

`sklearnsvm OneClassSVM`

`classsklearnsvm OneClassSVM kernel'rbf' degree3 gamma'autodeprecated' coef000`

`tol0001 nu05 shrinkingTrue cachesize200 verboseFalse`

`maxiter1 randomstateNone`

Unsupervised Outlier Detection

Estimate the support of a highdimensional distribution

The implementation is based on `libsvm`

Read more in the User Guide

Parameters

`kernel` string optional default'rbf' Specifies the kernel type to be used in the algorithm

It must be one of 'linear' 'poly' 'rbf' 'sigmoid' 'precomputed' or a callable If none is

given 'rbf' will be used If a callable is given it is used to precompute the kernel matrix

`degree` int optional default3 Degree of the polynomial kernel function 'poly' Ignored

by all other kernels

`gamma` float optional default'auto' Kernel coefficient for 'rbf' 'poly' and 'sigmoid'

Current default is 'auto' which uses $1/n$ features if `gamma`scale is passed then it

uses $1/n$ features `X`var as value of `gamma` The current default of `gamma` 'auto'

will change to 'scale' in version 022 'autodeprecated' a deprecated version of 'auto' is

used as a default indicating that no explicit value of `gamma` was passed

`coef0` float optional default00 Independent term in kernel function It is only significant

in 'poly' and 'sigmoid'

`tol`float optional Tolerance for stopping criterion

`nu`float optional An upper bound on the fraction of training errors and a lower bound of the

fraction of support vectors Should be in the interval 0 1 By default 05 will be taken

`shrinking` boolean optional Whether to use the shrinking heuristic

`cachesize` float optional Specify the size of the kernel cache in MB

637sklearnsvm Support Vector Machines 2315

scikitlearn user guide Release 0213

verbose bool default False Enable verbose output Note that this setting takes advantage of a perprocess runtime setting in libsvm that if enabled may not work properly in a multithreaded context

maxiter int optional default1 Hard limit on iterations within solver or 1 for no limit

randomstate int RandomState instance or None optional defaultNone Ignored

Deprecated since version 020 randomstate has been deprecated in 020 and will be removed in 022

Attributes

support arraylike shape nSV Indices of support vectors

supportvectors arraylike shape nSV nfeatures Support vectors

dualcoef array shape 1 nSV Coefficients of the support vectors in the decision function

coef array shape 1 nfeatures Weights assigned to the features coefficients in the primal problem This is only available in the case of a linear kernel

coef is readonly property derived from dualcoef andsupportvectors

intercept array shape 1 Constant in the decision function

offset float Offset used to define the decision function from the raw scores We have the relation decisionfunction scoresamples offset The offset is the opposite of intercept and is provided for consistency with other outlier detection algorithms

Examples

```
from sklearnsvm import OneClassSVM
X = 0 044 045 046 1
clf = OneClassSVMgammaautofitX
clf.predictX
array1 1 1 1 1
clfscoresamplesX
array17798 20547 20556 20561 17332
```

Methods

decisionfunction self X Signed distance to the separating hyperplane

fitself X y sampleweight Detects the soft boundary of the set of samples X

fitpredict self X y Performs fit on X and returns labels for X

getparams self deep Get parameters for this estimator

predict self X Perform classification on samples in X

scoresamples self X Raw scoring function of the samples

setparams self params Set the parameters of this estimator

init selfkernel'rbf' degree3 gamma'autodeprecated' coef000 tol0001 nu05 shrinkingTrue cachesize200 verboseFalse maxiter1 randomstateNone

decisionfunction selfX

Signed distance to the separating hyperplane

2316 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Signed distance is positive for an inlier and negative for an outlier

Parameters

Xarraylike shape nsamples nfeatures

Returns

dec arraylike shape nsamples Returns the decision function of the samples

fitselfXyNone sampleweightNone params

Detects the soft boundary of the set of samples X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Set of samples where nsamples is the number of samples and nfeatures is the number of features

sampleweight arraylike shape nsamples Persample weights Rescale C per sample

Higher weights force the classifier to put more emphasis on these points

yIgnored not used present for API (Application Programming Interface) consistency by convention

Returns

self object

Notes

If X is not a Cordered contiguous array it is copied

fitpredict selfXyNone

Performs fit on X and returns labels for X

Returns 1 for outliers and 1 for inliers

Parameters

Xndarray shape nsamples nfeatures Input data

yIgnored not used present for API (Application Programming Interface) consistency by convention

Returns

yndarray shape nsamples 1 for inliers 1 for outliers

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Perform classification on samples in X

For a oneclass model 1 or 1 is returned

Parameters

637sklearnsvm Support Vector Machines 2317

scikitlearn user guide Release 0213

Xarraylike sparse matrix shape nsamples nfeatures For kernel"precomputed"

the expected shape of X is nsamplestest nsamplestrain

Returns

ypred array shape nsamples Class labels for samples in X

scoresamples selfX

Raw scoring function of the samples

Parameters

Xarraylike shape nsamples nfeatures

Returns

scoresamples arraylike shape nsamples Returns the unshifted scoring function of the samples

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnsvmOneClassSVM

- Comparing anomaly detection algorithms for outlier detection on toy datasets

- Outlier detection on a real data set

- Species distribution modeling

- Libsvm GUI

- Oneclass SVM (Support Vector Machine) with nonlinear kernel RBF (Radial Basis Function)

sklearnsvm SVC (Support Vector Classification)

classssklearnsvm SVCC10 kernel'rbf' degree3 gamma'autodeprecated' coef000 shrink

ingTrue probabilityFalse tol0001 cachesize200 classweightNone

verboseFalse maxiter1 decisionfunctionshape'ovr' ran

domstateNone

Csupport Vector Classification

The implementation is based on libsvm The fit time scales at least quadratically with the number of samples and may be impractical beyond tens of thousands of samples For large datasets consider using sklearn

linearmodelLinearSVC orsklearnlinearmodelSGDClassifier instead possibly after

asklearnkernelapproximationNystroem transformer

The multiclass support is handled according to a onevsone scheme

For details on the precise mathematical formulation of the provided kernel functions and how gamma coef0

anddegree affect each other see the corresponding section in the narrative documentation Kernel functions

Read more in the User Guide

2318 Chapter 6 API (Application Programming Interface) Reference

Parameters

Cfloat optional default10 Penalty parameter C of the error term

kernel string optional default'rbf' Specifies the kernel type to be used in the algorithm
It must be one of 'linear' 'poly' 'rbf' 'sigmoid' 'precomputed' or a callable If none
is given 'rbf' will be used If a callable is given it is used to precompute the ker
nel matrix from data matrices that matrix should be an array of shape nsamples
nsamples

degree int optional default3 Degree of the polynomial kernel function 'poly' Ignored
by all other kernels

gamma float optional default'auto' Kernel coefficient for 'rbf' 'poly' and 'sigmoid'
Current default is 'auto' which uses 1 / nfeatures if gammascale is passed then it
uses 1 / nfeatures Xvar as value of gamma The current default of gamma 'auto'
will change to 'scale' in version 022 'autodeprecated' a deprecated version of 'auto' is
used as a default indicating that no explicit value of gamma was passed

coef0 float optional default00 Independent term in kernel function It is only significant
in 'poly' and 'sigmoid'

shrinking boolean optional defaultTrue Whether to use the shrinking heuristic

probability boolean optional defaultFalse Whether to enable probability estimates This
must be enabled prior to calling fit and will slow down that method

tolfloat optional default1e3 Tolerance for stopping criterion

cachesize float optional Specify the size of the kernel cache in MB

classweight dict 'balanced' optional Set the parameter C of class i to classweightiC
for SVC (Support Vector Classification) If not given all classes are supposed to have weight one The "balanced" mode
uses the values of y to automatically adjust weights inversely proportional to class frequen
cies in the input data as nsamples / nclasses npbincounty

verbose bool default False Enable verbose output Note that this setting takes advantage
of a perprocess runtime setting in libsvm that if enabled may not work properly in a
multithreaded context

maxiter int optional default1 Hard limit on iterations within solver or 1 for no limit

decisionfunctionshape 'ovo' 'ovr' default'ovr' Whether to return a onevsrest 'ovr'
decision function of shape nsamples nclasses as all other classifiers or the original
onevsone 'ovo' decision function of libsvm which has shape nsamples nclasses
nclasses 1 2 However onevsone 'ovo' is always used as multiclass strategy
Changed in version 019 decisionfunctionshape is 'ovr' by default
New in version 017 decisionfunctionshape'ovr' is recommended
Changed in version 017 Deprecated decisionfunctionshape'ovo' and None

randomstate int RandomState instance or None optional defaultNone The seed of the
pseudo random number generator used when shuffling the data for probability estimates
If int randomstate is the seed used by the random number generator If RandomState
instance randomstate is the random number generator If None the random number gen
erator is the RandomState instance used by nprandom

Attributes

support arraylike shape nSV Indices of support vectors

637sklearnsvm Support Vector Machines 2319

scikitlearn user guide Release 0213

supportvectors arraylike shape nSV nfeatures Support vectors

nsupport arraylike dtypeint32 shape nclass Number of support vectors for each class

dualcoef array shape nclass1 nSV Coefficients of the support vector in the decision function For multiclass coefficient for all 1vs1 classifiers The layout of the coefficients in the multiclass case is somewhat nontrivial See the section about multiclass classification in the SVM (Support Vector Machine) section of the User Guide for details

coef array shape nclass nclass1 2 nfeatures Weights assigned to the features

coefficients in the primal problem This is only available in the case of a linear kernel

coef is a readonly property derived from dualcoef andsupportvectors

intercept array shape nclass nclass1 2 Constants in decision function

fitstatus int 0 if correctly fitted 1 otherwise will raise warning

probA array shape nclass nclass1 2

probB array shape nclass nclass1 2 If probabilityTrue the parameters

learned in Platt scaling to produce probability estimates from decision values If prob

abilityFalse an empty array Platt scaling uses the logistic function 1 1

expdecisionvalue probA probB whereprobA andprobB are

learned from the dataset R20c70293ef722 For more information on the multiclass case

and training procedure see section 8 of R20c70293ef721

See also

SVR (Support Vector Regression) Support Vector Machine for Regression implemented using libsvm

LinearSVC Scalable Linear Support Vector Machine for classification implemented using liblinear Check

the See also section of LinearSVC for more comparison element

References

R20c70293ef721 R20c70293ef722

Examples

```
import numpy as np
```

```
X nparray1 1 2 1 1 1 2 1
```

```
y nparray1 1 2 2
```

```
from sklearnsvm import SVC (Support Vector Classification)
```

```
clf SVCgammaauto
```

```
clffitX y
```

```
SVCC10 cachesize200 classweightNone coef000
```

```
decisionfunctionshapeovr degree3 gammaauto kernelrbf
```

```
maxiter1 probabilityFalse randomstateNone shrinkingTrue
```

```
tol0001 verboseFalse
```

```
printclfpredict08 1
```

```
1
```

2320 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Methods

decisionfunction self X Evaluates the decision function for the samples in X
fitself X y sampleweight Fit the SVM (Support Vector Machine) model according to the given training data
getparams self deep Get parameters for this estimator
predict self X Perform classification on samples in X
score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator
init selfC10 kernel'rbf' degree3 gamma'autodeprecated' coef000 shrink
ingTrue probabilityFalse tol0001 cachesize200 classweightNone ver
boseFalse maxiter1 decisionfunctionshape'ovr' randomstateNone
decisionfunction selfX

Evaluates the decision function for the samples in X

Parameters

Xarraylike shape nsamples nfeatures

Returns

Xarraylike shape nsamples nclasses nclasses1 2 Returns the decision func
tion of the sample for each class in the model If decisionfunctionshape'ovr' the shape
is nsamples nclasses

Notes

If decisionfunctionshape'ovo' the function values are proportional to the distance of the samples X to
the separating hyperplane If the exact distances are required divide the function values by the norm of
the weight vector coef See also this question for further details If decisionfunctionshape'ovr'
the decision function is a monotonic transformation of ovo decision function

fitselfXysampleweightNone

Fit the SVM (Support Vector Machine) model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where
nsamples is the number of samples and nfeatures is the number of features For ker
nel"precomputed" the expected shape of X is nsamples nsamples
yarraylike shape nsamples Target values class labels in classification real numbers
in regression

sampleweight arraylike shape nsamples Persample weights Rescale C per sample
Higher weights force the classifier to put more emphasis on these points

Returns

self object

Notes

If X and y are not Cordered and contiguous arrays of npfloat64 and X is not a scipyparsesrmatrix X
andor y may be copied

637sklearnsvm Support Vector Machines 2321

scikitlearn user guide Release 0213

If X is a dense array then the other methods will not support sparse matrices as input

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfX

Perform classification on samples in X

For an oneclass model 1 or 1 is returned

Parameters

Xarraylike sparse matrix shape nsamples nfeatures For kernel"precomputed"

the expected shape of X is nsamplestest nsamplestrain

Returns

ypred array shape nsamples Class labels for samples in X

predictlogproba

Compute log probabilities of possible outcomes for samples in X

The model need to have probability information computed at training time fit with attribute

probability set to True

Parameters

Xarraylike shape nsamples nfeatures For kernel"precomputed" the expected

shape of X is nsamplestest nsamplestrain

Returns

Tarraylike shape nsamples nclasses Returns the logprobabilities of the sample for

each class in the model The columns correspond to the classes in sorted order as they

appear in the attribute classes

Notes

The probability model is created using cross validation so the results can be slightly different than those

obtained by predict Also it will produce meaningless results on very small datasets

predictproba

Compute probabilities of possible outcomes for samples in X

The model need to have probability information computed at training time fit with attribute

probability set to True

Parameters

Xarraylike shape nsamples nfeatures For kernel"precomputed" the expected

shape of X is nsamplestest nsamplestrain

Returns

2322 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Tarraylike shape nsamples nclasses Returns the probability of the sample for each class in the model The columns correspond to the classes in sorted order as they appear in the attribute classes

Notes

The probability model is created using cross validation so the results can be slightly different than those obtained by predict Also it will produce meaningless results on very small datasets

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearnsvmSVC

- Multilabel classification
 - Explicit feature map approximation for RBF (Radial Basis Function) kernels
 - Faces recognition example using eigenfaces and SVMs
 - Libsvm GUI
 - Recognizing handwritten digits
 - Plot classification probability
 - Classifier comparison
 - Concatenating multiple feature extraction methods
 - Plot the decision boundaries of a VotingClassifier
 - Crossvalidation on Digits Dataset Exercise
 - SVM (Support Vector Machine) Exercise
 - Recursive feature elimination
- 637sklearnsvm Support Vector Machines 2323

scikitlearn user guide Release 0213

- Recursive feature elimination with crossvalidation
 - Test with permutations the significance of a classification score
 - Univariate Feature Selection
 - Plotting Validation Curves
 - Parameter estimation using grid search with crossvalidation
 - Receiver Operating Characteristic ROC with cross validation
 - Nested versus nonnested crossvalidation
 - Confusion matrix
 - Receiver Operating Characteristic ROC
 - Plotting Learning Curves
 - Feature discretization
 - Decision boundary of label propagation versus SVM (Support Vector Machine) on the Iris dataset
 - SVM (Support Vector Machine) Maximum margin separating hyperplane
 - SVM (Support Vector Machine) with custom kernel
 - SVM (Support Vector Machine) Weighted samples
 - SVM (Support Vector Machine) Separating hyperplane for unbalanced classes
 - SVMKernels
 - SVMAnova SVM (Support Vector Machine) with univariate feature selection
 - SVM (Support Vector Machine) Margins Example
 - Plot different SVM (Support Vector Machine) classifiers in the iris dataset
 - RBF (Radial Basis Function) SVM (Support Vector Machine) parameters
- sklearnsvm SVR (Support Vector Regression)
- classssklearnsvm SVRkernel'rbf' degree3 gamma'autodeprecated' coef000 tol0001
- C10 epsilon01 shrinkingTrue cachesize200 verboseFalse
- maxiter1

EpsilonSupport Vector Regression

The free parameters in the model are C and epsilon

The implementation is based on libsvm The fit time complexity is more than quadratic with the number of samples which makes it hard to scale to datasets with more than a couple of 10000 samples For large datasets consider using sklearnlinearmodelLinearSVR orsklearnlinearmodelSGDRegressor instead possibly after a sklearnkernelapproximationNystroem transformer

Read more in the User Guide

Parameters

kernel string optional default'rbf' Specifies the kernel type to be used in the algorithm It must be one of 'linear' 'poly' 'rbf' 'sigmoid' 'precomputed' or a callable If none is given 'rbf' will be used If a callable is given it is used to precompute the kernel matrix

degree int optional default3 Degree of the polynomial kernel function 'poly' Ignored by all other kernels

2324 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

gamma float optional default'auto' Kernel coefficient for 'rbf' 'poly' and 'sigmoid'

Current default is 'auto' which uses 1 / nfeatures if gammascale is passed then it uses 1 / nfeatures Xvar as value of gamma The current default of gamma 'auto' will change to 'scale' in version 022 'autodeprecated' a deprecated version of 'auto' is used as a default indicating that no explicit value of gamma was passed

coef0 float optional default00 Independent term in kernel function It is only significant in 'poly' and 'sigmoid'

tolfloat optional default1e3 Tolerance for stopping criterion

Cfloat optional default10 Penalty parameter C of the error term

epsilon float optional default01 Epsilon in the epsilonSVR model It specifies the epsilon-tube within which no penalty is associated in the training loss function with points predicted within a distance epsilon from the actual value

shrinking boolean optional defaultTrue Whether to use the shrinking heuristic

cachesize float optional Specify the size of the kernel cache in MB

verbose bool default False Enable verbose output Note that this setting takes advantage of a perprocess runtime setting in libsvm that if enabled may not work properly in a multithreaded context

maxiter int optional default1 Hard limit on iterations within solver or 1 for no limit

Attributes

support arraylike shape (nSV) Indices of support vectors

supportvectors arraylike shape (nSV, nfeatures) Support vectors

dualcoef array shape (1, nSV) Coefficients of the support vector in the decision function

coef array shape (1, nfeatures) Weights assigned to the features coefficients in the primal problem This is only available in the case of a linear kernel

coef is readonly property derived from dualcoef and supportvectors

intercept array shape (1) Constants in decision function

See also

NuSVR Support Vector Machine for regression implemented using libsvm using a parameter to control the number of support vectors

LinearSVR Scalable Linear Support Vector Machine for regression implemented using liblinear

Notes

References LIBSVM A Library for Support Vector Machines

Examples

```
from sklearnsvm import SVR (Support Vector Regression)
import numpy as np
nsamples nfeatures 10 5
rng np.random.RandomState(0)
637sklearnsvm Support Vector Machines 2325
```

scikitlearn user guide Release 0213

y rnggrandnnsamples

X rnggrandnnsamples nfeatures

clf SVRgamma C10 epsilon02

clffitX y

SVRC10 cachesize200 coef000 degree3 epsilon02 gamma C10

kernelrbf maxiter1 shrinkingTrue tol0001 verboseFalse

Methods

fitself X y sampleweight Fit the SVM (Support Vector Machine) model according to the given training data

getparams self deep Get parameters for this estimator

predict self X Perform regression on samples in X

score self X y sampleweight Returns the coefficient of determination R2 of the pre

diction

setparams self params Set the parameters of this estimator

init selfkernel'rbf' degree3 gamma'autodeprecated' coef000 tol0001 C10 ep

silon01 shrinkingTrue cachesize200 verboseFalse maxiter1

fitselfXysampleweightNone

Fit the SVM (Support Vector Machine) model according to the given training data

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Training vectors where

nsamples is the number of samples and nfeatures is the number of features For ker

nel"precomputed" the expected shape of X is nsamples nsamples

yarraylike shape nsamples Target values class labels in classification real numbers

in regression

sampleweight arraylike shape nsamples Persample weights Rescale C per sample

Higher weights force the classifier to put more emphasis on these points

Returns

self object

Notes

If X and y are not Cordered and contiguous arrays of npfloat64 and X is not a scipysparsecsrmatrix X

andor y may be copied

If X is a dense array then the other methods will not support sparse matrices as input

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

2326 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

predictselfX

Perform regression on samples in X

For an oneclass model 1 inlier or 1 outlier is returned

Parameters

Xarraylike sparse matrix shape nsamples nfeatures For kernel”precomputed”

the expected shape of X is nsamplestest nsamplestrain

Returns

ypred array shape nsamples

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{truemean}$ $2sum$ The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it’s possible to update each component

of a nested object

Returns

self

Examples using sklearnsvmSVR

•Comparison of kernel ridge regression and SVR (Support Vector Regression)

•Prediction Latency

637sklearnsvm Support Vector Machines 2327

scikitlearn user guide Release 0213

•Support Vector Regression SVR (Support Vector Regression) using linear and nonlinear kernels
svml1minc X y loss fitintercept Return the lowest bound for C such that for C in l1minC
infinity the model is guaranteed not to be empty

sklearnsvm l1minc

sklearnsvm l1minc Xyloss’squaredhinge’ fitinterceptTrue interceptscaling10

Return the lowest bound for C such that for C in l1minC infinity the model is guaranteed not
to be empty This applies to l1 penalized classifiers such as LinearSVC with penalty’l1’ and lin
earmodelLogisticRegression with penalty’l1’

This value is valid if classweight parameter in fit is not set

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures Training vector where
nsamples in the number of samples and nfeatures is the number of features

yarray shape nsamples Target vector relative to X

loss ‘squaredhinge’ ‘log’ default ‘squaredhinge’ Specifies the loss function With
‘squaredhinge’ it is the squared hinge loss aka L2 loss With ‘log’ it is the loss of
logistic regression models

fitintercept bool default True Specifies if the intercept should be fitted by the model It
must match the fit method parameter

interceptscaling float default 1 when fitintercept is True instance vector x becomes x
interceptscaling ie a “synthetic” feature with constant value equals to interceptscaling
is appended to the instance vector It must match the fit method parameter

Returns

l1minc float minimum value for C

Examples using sklearnsvml1minc

•Regularization path of L1 Logistic Regression

6372 Lowlevel methods

svmlibsvmcrossvalidation Binding of the crossvalidation routine lowlevel routine

svmlibsvmdecisionfunction Predict margin libsvm name for this is predictvalues

svmlibsvmfit Train the model using libsvm lowlevel method

svmlibsvmpredict Predict target values of X given a model lowlevel method

svmlibsvmpredictproba Predict probabilities

sklearnsvmlibsvm crossvalidation

sklearnsvmlibsvm crossvalidation

Binding of the crossvalidation routine lowlevel routine

Parameters

2328 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Xarraylike dtypefloat sizensamples nfeatures

Yarray dtypefloat sizensamples target vector

svmtypes 0 1 2 3 4 Type of SVM (Support Vector Machine) C SVC (Support Vector Classification) nu SVC (Support Vector Classification) one class ep

kernel 'linear' 'rbf' 'poly' 'sigmoid' 'precomputed' Kernel to use in the model linear

polynomial RBF (Radial Basis Function) sigmoid or precomputed

degree int Degree of the polynomial kernel only relevant if kernel is set to polynomial

gamma float Gamma parameter in rbf poly and sigmoid kernels Ignored by other kernels

01 by default

coef0 float Independent parameter in polysigmoid kernel

tolfloat Stopping criteria

Cfloat C parameter in CSupport Vector Classification

nufloat

cache size float

randomseed int optional Seed for the random number generator used for probability esti

mates 0 by default

Returns

target array float

sklearnsvmlibsvm decisionfunction

sklearnsvmlibsvm decisionfunction

Predict margin libsvm name for this is predictvalues

We have to reconstruct model and parameters to make sure we stay in sync with the python object

sklearnsvmlibsvm fit

sklearnsvmlibsvm fit

Train the model using libsvm lowlevel method

Parameters

Xarraylike dtypefloat64 sizensamples nfeatures

Yarray dtypefloat64 sizensamples target vector

svmtypes 0 1 2 3 4 optional Type of SVM (Support Vector Machine) CSVC NuSVC OneClassSVM Ep

silonSVR or NuSVR respectively 0 by default

kernel 'linear' 'rbf' 'poly' 'sigmoid' 'precomputed' optional Kernel to use in the model

linear polynomial RBF (Radial Basis Function) sigmoid or precomputed 'rbf' by default

degree int32 optional Degree of the polynomial kernel only relevant if kernel is set to poly

nomial 3 by default

gamma float64 optional Gamma parameter in rbf poly and sigmoid kernels Ignored by

other kernels 01 by default

coef0 float64 optional Independent parameter in polysigmoid kernel 0 by default

tolfloat64 optional Numeric stopping criterion WRITE ME 1e3 by default

637sklearnsvm Support Vector Machines 2329

scikitlearn user guide Release 0213

Cfloat64 optional C parameter in CSupport Vector Classification 1 by default

nufloat64 optional 05 by default

epsilon double optional 01 by default

classweight array dtype float64 shape nclasses optional npempty0 by default

sampleweight array dtype float64 shape nsamples optional npempty0 by default

shrinking int optional 1 by default

probability int optional 0 by default

cachesize float64 optional Cache size for gram matrix columns in megabytes 100 by

default

maxiter int 1 for no limit optional Stop solver after this many iterations regardless of

accuracy XXX Currently there is no API (Application Programming Interface) to know whether this kicked in 1 by default

randomseed int optional Seed for the random number generator used for probability esti

mates 0 by default

Returns

support array shapensupport index of support vectors

supportvectors array shapensupport nfeatures support vectors equivalent to

Xsupport Will return an empty array in the case of precomputed kernel

nclassSV array number of support vectors in each class

svcoef array coefficients of support vectors in decision function

intercept array intercept in decision function

probA probB array probability estimates empty array for probabilityFalse

sklearnsvmlibsvm predict

sklearnsvmlibsvm predict

Predict target values of X given a model lowlevel method

Parameters

Xarraylike dtypefloat sizensamples nfeatures

svmtype 0 1 2 3 4 Type of SVM (Support Vector Machine) C SVC (Support Vector Classification) nu SVC (Support Vector Classification) one class ep

kernel 'linear' 'rbf' 'poly' 'sigmoid' 'precomputed' Type of kernel

degree int Degree of the polynomial kernel

gamma float Gamma parameter in rbf poly and sigmoid kernels Ignored by other kernels

01 by default

coef0 float Independent parameter in polysigmoid kernel

Returns

decvalues array predicted values

2330 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
sklearnsvmlibsvm predictproba
sklearnsvmlibsvm predictproba
Predict probabilities
svmmodel stores all parameters needed to predict a given value
For speed all real work is done at the C level in function copypredict libsvmhelperc
We have to reconstruct model and parameters to make sure we stay in sync with the python object
See sklearnsvmpredict for a complete list of parameters
Parameters
Xarraylike dtypefloat
kernel 'linear' 'rbf' 'poly' 'sigmoid' 'precomputed'
Returns
decvalues array predicted values
638sklearnntree Decision Trees
The sklearnntree module includes decision treebased models for classification and regression
User guide See the Decision Trees section for further details
treeDecisionTreeClassifier criterion A decision tree classifier
treeDecisionTreeRegressor criterion A decision tree regressor
treeExtraTreeClassifier criterion An extremely randomized tree classifier
treeExtraTreeRegressor criterion An extremely randomized tree regressor
6381sklearnntree DecisionTreeClassifier
class sklearnntree DecisionTreeClassifier criterion'gini' splitter'best' maxdepthNone
minsamplessplit2 minsamplesleaf1
minweightfractionleaf00
maxfeaturesNone ran
domstateNone maxleafnodesNone
minimpuritydecrease00
minimpuritysplitNone classweightNone
presortFalse
A decision tree classifier
Read more in the User Guide
Parameters
criterion string optional default"gini" The function to measure the quality of a split Sup
ported criteria are "gini" for the Gini impurity and "entropy" for the information gain
splitter string optional default"best" The strategy used to choose the split at each node
Supported strategies are "best" to choose the best split and "random" to choose the best
random split
maxdepth int or None optional defaultNone The maximum depth of the tree If None
638sklearnntree Decision Trees 2331

scikitlearn user guide Release 0213

then nodes are expanded until all leaves are pure or until all leaves contain less than minsamplesplit samples

minsamplesplit int float optional default2 The minimum number of samples required to split an internal node

- If int then consider minsamplesplit as the minimum number
- If float then minsamplesplit is a fraction and ceilminsamplesplit nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions

minsamplesleaf int float optional default1 The minimum number of samples required to be at a leaf node A split point at any depth will only be considered if it leaves at least minsamplesleaf training samples in each of the left and right branches This may have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number
- If float then minsamplesleaf is a fraction and ceilminsamplesleaf nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the sum total of weights of all the input samples required to be at a leaf node Samples have equal weight when sampleweight is not provided

maxfeatures int float string or None optional defaultNone The number of features to consider when looking for the best split

- If int then consider maxfeatures features at each split
- If float then maxfeatures is a fraction and intmaxfeatures nfeatures features are considered at each split
- If "auto" then maxfeaturessqrtnfeatures
- If "sqrt" then maxfeaturesqrtnfeatures
- If "log2" then maxfeatureslog2nfeatures
- If None then maxfeaturesnfeatures

Note the search for a split does not stop until at least one valid partition of the node samples is found even if it requires to effectively inspect more than maxfeatures features

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

maxleafnodes int or None optional defaultNone Grow a tree with maxleafnodes

in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then unlimited number of leaf nodes

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$N_t \cdot N \cdot \text{impurity} - N_t R \cdot N_t \cdot \text{rightimpurity}$

$- N_t L \cdot N_t \cdot \text{leftimpurity}$

2332 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

where N is the total number of samples N_t is the number of samples at the current node
 N_L is the number of samples in the left child and N_R is the number of samples in the right child

NN_tN_R and N_L all refer to the weighted sum if `sampleweight` is passed
New in version 019

`minimpuritysplit` float default $1e7$ Threshold for early stopping in tree growth A node will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 `minimpuritysplit` has been deprecated in favor of `minimpuritydecrease` in 019 The default value of `minimpuritysplit` will change from $1e7$ to 0 in 023 and it will be removed in 025 Use `minimpuritydecrease` instead

`classweight` dict list of dicts “balanced” or None default None Weights associated with classes in the form `classlabel weight` If not given all classes are supposed to have weight one For multioutput problems a list of dicts can be provided in the same order as the columns of `y`

Note that for multioutput including multilabel weights should be defined for each class of every column in its own dict For example for fourclass multilabel classification weights should be 0 1 1 1 0 1 1 5 0 1 1 1 0 1 1 1 instead of 11 25 31 41

The “balanced” mode uses the values of `y` to automatically adjust weights inversely proportional to class frequencies in the input data as $n_{\text{samples}} / n_{\text{classes}}$ np

`bincounty`
For multioutput the weights of each column of `y` will be multiplied

Note that these weights will be multiplied with `sampleweight` passed through the fit method if `sampleweight` is specified

`presort` bool optional default False Whether to presort the data to speed up the finding of best splits in fitting For the default settings of a decision tree on large datasets setting this to true may slow down the training process When using either a smaller dataset or a restricted depth this may speed up the training

Attributes

`classes` array of shape `n_classes` or a list of such arrays The classes labels single output problem or a list of arrays of class labels multioutput problem

`featureimportances` array of shape `n_features` Return the feature importances

`maxfeatures` int The inferred value of `maxfeatures`

`n_classes` int or list The number of classes for single output problems or a list containing the number of classes for each output for multioutput problems

`n_features` int The number of features when fit is performed

`n_outputs` int The number of outputs when fit is performed

`tree` Tree object The underlying Tree object Please refer to `helpsklearn.tree`

`treeTree` for attributes of Tree object and Understanding the decision tree structure for basic usage of these attributes

See also

`DecisionTreeRegressor`

638sklearn.tree Decision Trees 2333

Notes

The default values for the parameters controlling the size of the trees eg maxdepth minsamplesleaf etc lead to fully grown and unpruned trees which can potentially be very large on some data sets To reduce memory consumption the complexity and size of the trees should be controlled by setting those parameter values

The features are always randomly permuted at each split Therefore the best found split may vary even with the same training data and maxfeaturesnfeatures if the improvement of the criterion is identical for several splits enumerated during the search of the best split To obtain a deterministic behaviour during fitting randomstate has to be fixed

References

Rb1ec977cd3071 Rb1ec977cd3072 Rb1ec977cd3073 Rb1ec977cd3074

Examples

```
from sklearn.datasets import loadiris
from sklearn.model_selection import cross_val_score
from sklearn.tree import DecisionTreeClassifier
clf = DecisionTreeClassifier(random_state=0)
iris = loadiris()
cross_val_score(clf, iris.data, iris.target, cv=10)
```

array 1 093 086 093 093
093 093 1 093 1

Methods

- apply self X checkinput Returns the index of the leaf that each sample is predicted as
 - decisionpath self X checkinput Return the decision path in the tree
 - fitself X y sampleweight Build a decision tree classifier from the training set X y
 - getdepth self Returns the depth of the decision tree
 - getnleaves self Returns the number of leaves of the decision tree
 - getparams self deep Get parameters for this estimator
 - predict self X checkinput Predict class or regression value for X
 - predictlogproba self X Predict class logprobabilities of the input samples X
 - predictproba self X checkinput Predict class probabilities of the input samples X
 - score self X y sampleweight Returns the mean accuracy on the given test data and labels
 - setparams self params Set the parameters of this estimator
 - init selfcriterion'gini' splitter'best' maxdepthNone minsamplessplit2 minsamplesleaf1 minweightfractionleaf00 maxfeaturesNone randomstateNone maxleafnodesNone minimpuritydecrease00 minimpuritysplitNone classweightNone presortFalse
- 2334 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

applyselfXcheckinputTrue

Returns the index of the leaf that each sample is predicted as

New in version 017

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

Xleaves arraylike shape nsamples For each datapoint x in X return the index of the leaf x ends up in Leaves are numbered within 0 selftree nodecount possibly with gaps in the numbering

decisionpath selfXcheckinputTrue

Return the decision path in the tree

New in version 018

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

indicator sparse csr array shape nsamples nnodes Return a node indicator matrix where non zero elements indicates that the samples goes through the nodes

featureimportances

Return the feature importances

The importance of a feature is computed as the normalized total reduction of the criterion brought by that

feature It is also known as the Gini importance

Returns

featureimportances array shape nfeatures

fitselfXysampleweightNone checkinputTrue XidxsortedNone

Build a decision tree classifier from the training set X y

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The training input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsc matrix

yarraylike shape nsamples or nsamples noutputs The target values class labels as integers or strings

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Splits that would create child nodes with net zero or negative weight are ignored while searching for a split in each node Splits are also ignored if

they would result in any single class carrying a negative weight in either child node

638sklearn.tree Decision Trees 2335

scikitlearn user guide Release 0213

checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Xidxsorted arraylike shape nsamples nfeatures optional The indexes of the sorted training input samples If many tree are grown on the same dataset this allows the ordering to be cached between trees If None the data will be sorted here Don't use this parameter unless you know what to do

Returns

self object

getdepth self

Returns the depth of the decision tree

The depth of a tree is the maximum distance between the root and any leaf

getnleaves self

Returns the number of leaves of the decision tree

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXcheckinputTrue

Predict class or regression value for X

For a classification model the predicted class for each sample in X is returned For a regression model the predicted value based on X is returned

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

yarray of shape nsamples or nsamples noutputs The predicted classes or the predict values

predictlogproba selfX

Predict class logprobabilities of the input samples X

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

Returns

2336 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

array of shape (n_samples, n_classes) or a list of n_outputs such arrays if n_outputs

1 The class logprobabilities of the input samples The order of the classes corresponds to that in the attribute classes

predict_proba self.Xcheckinput=True

Predict class probabilities of the input samples X

The predicted class probability is the fraction of samples of the same class in a leaf

checkinput boolean default=True Allow to bypass several input checking Don't use this parameter unless you know what you do

Parameters

X arraylike or sparse matrix of shape (n_samples, n_features) The input samples Internally it will be converted to dtype np.float32 and if a sparse matrix is provided to

a sparse matrix

checkinput bool Run checkarray on X

Returns

array of shape (n_samples, n_classes) or a list of n_outputs such arrays if n_outputs

1 The class probabilities of the input samples The order of the classes corresponds to that in the attribute classes

score self.Xysampleweight=None

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

X arraylike shape (n_samples, n_features) Test samples

y arraylike shape (n_samples, n_outputs) True labels for X

sampleweight arraylike shape (n_samples, optional) Sample weights

Returns

score float Mean accuracy of self.predict_X wrt y

set_params self.params

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form component.parameter so that it's possible to update each component of a nested object

Returns

self

Examples using sklearn.tree.DecisionTreeClassifier

- Classifier comparison
 - Plot the decision boundaries of a VotingClassifier
 - Twoclass AdaBoost
 - Multiclass AdaBoosted Decision Trees
- 638 sklearn.tree Decision Trees 2337

scikitlearn user guide Release 0213

- Discrete versus Real AdaBoost
- Plot the decision surfaces of ensembles of trees on the iris dataset
- Demonstration of multimetric evaluation on crossvalscore and GridSearchCV
- Plot the decision surface of a decision tree on the iris dataset
- Understanding the decision tree structure

6382sklearn DecisionTreeRegressor
class sklearn DecisionTreeRegressor criterion='mse' splitter='best' maxdepth=None
minsamplesplit2 minsamplesleaf1
minweightfractionleaf00 maxfeaturesNone
randomstateNone maxleafnodesNone
minimpuritydecrease00
minimpuritysplitNone presortFalse

A decision tree regressor
Read more in the User Guide

Parameters
criterion string optional default"mse" The function to measure the quality of a split Supported criteria are "mse" for the mean squared error which is equal to variance reduction as feature selection criterion and minimizes the L2 loss using the mean of each terminal node "friedmanmse" which uses mean squared error with Friedman's improvement score for potential splits and "mae" for the mean absolute error which minimizes the L1 loss using the median of each terminal node

New in version 018 Mean Absolute Error MAE criterion
splitter string optional default"best" The strategy used to choose the split at each node Supported strategies are "best" to choose the best split and "random" to choose the best random split

maxdepth int or None optional defaultNone The maximum depth of the tree If None then nodes are expanded until all leaves are pure or until all leaves contain less than minsamplesplit samples

minsamplesplit int float optional default2 The minimum number of samples required to split an internal node

- If int then consider minsamplesplit as the minimum number
- If float then minsamplesplit is a fraction and ceilminsamplesplit

nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions
minsamplesleaf int float optional default1 The minimum number of samples required to be at a leaf node A split point at any depth will only be considered if it leaves at least minsamplesleaf training samples in each of the left and right branches This may have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number
- If float then minsamplesleaf is a fraction and ceilminsamplesleaf

nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions
2338 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

minweightfractionleaf float optional default0 The minimum weighted fraction of the sum total of weights of all the input samples required to be at a leaf node Samples have equal weight when sampleweight is not provided

maxfeatures int float string or None optional defaultNone The number of features to consider when looking for the best split

- If int then consider maxfeatures features at each split
- If float then maxfeatures is a fraction and intmaxfeatures

nfeatures features are considered at each split

- If “auto” then maxfeaturesnfeatures
- If “sqrt” then maxfeaturesqrtnfeatures
- If “log2” then maxfeatureslog2nfeatures
- If None then maxfeaturesnfeatures

Note the search for a split does not stop until at least one valid partition of the node samples is found even if it requires to effectively inspect more than maxfeatures features

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

maxleafnodes int or None optional defaultNone Grow a tree with maxleafnodes

in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then unlimited number of leaf nodes

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$N_t \cdot N \cdot \text{impurity} - N_t R \cdot N_t \text{rightimpurity}$$

$$- N_t L \cdot N_t \text{leftimpurity}$$

whereNis the total number of samples Nt is the number of samples at the current node

NtL is the number of samples in the left child and NtR is the number of samples in the right child

NNtNtR andNtL all refer to the weighted sum if sampleweight is passed

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of

minimpuritydecrease in 019 The default value of minimpuritysplit

will change from 1e7 to 0 in 023 and it will be removed in 025 Use

minimpuritydecrease instead

presort bool optional defaultFalse Whether to presort the data to speed up the finding of best splits in fitting For the default settings of a decision tree on large datasets setting

this to true may slow down the training process When using either a smaller dataset or a restricted depth this may speed up the training

Attributes

featureimportances array of shape nfeatures Return the feature importances

638sklearn.tree Decision Trees 2339

scikitlearn user guide Release 0213
maxfeatures int The inferred value of maxfeatures
nfeatures int The number of features when fit is performed
noutputs int The number of outputs when fit is performed
tree Tree object The underlying Tree object Please refer to helpsklearn tree
treeTree for attributes of Tree object and Understanding the decision tree structure
for basic usage of these attributes
See also

DecisionTreeClassifier

Notes

The default values for the parameters controlling the size of the trees eg maxdepth
minsamplesleaf etc lead to fully grown and unpruned trees which can potentially be very large on
some data sets To reduce memory consumption the complexity and size of the trees should be controlled by
setting those parameter values
The features are always randomly permuted at each split Therefore the best found split may vary even with
the same training data and maxfeaturesnfeatures if the improvement of the criterion is identical for
several splits enumerated during the search of the best split To obtain a deterministic behaviour during fitting
randomstate has to be fixed

References

Ra37b7e3adb191 Ra37b7e3adb192 Ra37b7e3adb193 Ra37b7e3adb194

Examples

```
from sklearn.datasets import loadboston
from sklearn.model_selection import cross_val_score
from sklearn.tree import DecisionTreeRegressor
boston = loadboston
regressor = DecisionTreeRegressor(random_state=0)
cross_val_score(regressor, boston.data, boston.target, cv=10)
```

array [0.61 0.57 0.34 0.41 0.75
0.07 0.29 0.33 0.14 0.17]

Methods

apply self X checkinput Returns the index of the leaf that each sample is pre
dicted as
decisionpath self X checkinput Return the decision path in the tree
fitself X y sampleweight Build a decision tree regressor from the training set X
y
getdepth self Returns the depth of the decision tree

Continued on next page

getnleaves self Returns the number of leaves of the decision tree
getparams self deep Get parameters for this estimator
predict self X checkinput Predict class or regression value for X
score self X y sampleweight Returns the coefficient of determination R2 of the prediction
setparams self params Set the parameters of this estimator
init selfcriterion'mse' splitter'best' maxdepthNone minsamplesplit2
minsamplesleaf1 minweightfractionleaf00 maxfeaturesNone
randomstateNone maxleafnodesNone minimpuritydecrease00
minimpuritysplitNone presortFalse
applyselfXcheckinputTrue

Returns the index of the leaf that each sample is predicted as

New in version 017

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix
checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

Xleaves arraylike shape nsamples For each datapoint x in X return the index of the leaf x ends up in Leaves are numbered within 0 selftreenodecount possibly with gaps in the numbering

decisionpath selfXcheckinputTrue

Return the decision path in the tree

New in version 018

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix
checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

indicator sparse csr array shape nsamples nnodes Return a node indicator matrix where non zero elements indicates that the samples goes through the nodes

featureimportances

Return the feature importances

The importance of a feature is computed as the normalized total reduction of the criterion brought by that feature It is also known as the Gini importance

Returns

featureimportances array shape nfeatures

scikitlearn user guide Release 0213

fitselfXysampleweightNone checkinputTrue XidxsortedNone

Build a decision tree regressor from the training set X y

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The training input samples

Internally it will be converted to dtypefloat32 and if a sparse matrix is provided

to a sparsematrix

yarraylike shape nsamples noutputs The target values real num

bers Useddtypefloat64 andorderC for maximum efficiency

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Splits that would create child nodes with net zero or nega

tive weight are ignored while searching for a split in each node

checkinput boolean defaultTrue Allow to bypass several input checking Don't use

this parameter unless you know what you do

Xidxsorted arraylike shape nsamples nfeatures optional The indexes of the

sorted training input samples If many tree are grown on the same dataset this allows the

ordering to be cached between trees If None the data will be sorted here Don't use this

parameter unless you know what to do

Returns

self object

getdepth self

Returns the depth of the decision tree

The depth of a tree is the maximum distance between the root and any leaf

getnleaves self

Returns the number of leaves of the decision tree

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXcheckinputTrue

Predict class or regression value for X

For a classification model the predicted class for each sample in X is returned For a regression model

the predicted value based on X is returned

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Inter

nally it will be converted to dtypefloat32 and if a sparse matrix is provided to

a sparsematrix

checkinput boolean defaultTrue Allow to bypass several input checking Don't use

this parameter unless you know what you do

Returns

2342 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

array of shape nsamples or nsamples noutputs The predicted classes or the predict values

scoreselfXysampleweightNone

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

2sum and v is the total sum of squares $y_{true} - y_{truemean}$ 2sum The best possible score

is 10 and it can be negative because the model can be arbitrarily worse A constant model that always

predicts the expected value of y disregarding the input features would get a R2 score of 00

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where

nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of selfpredictX wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 023 to keep consistent with metricsr2score This will influence the score

method of all the multioutput regressors except for multioutputMultiOutputRegressor

To specify the default value manually and avoid the warning please either call metricsr2score

directly or make a custom scorer with metricsmakescorer the builtin scorer r2 uses

multioutputuniformaverage

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have

parameters of the form componentparameter so that it's possible to update each component

of a nested object

Returns

self

Examples using sklearnDecisionTreeRegressor

•Decision Tree Regression with AdaBoost

•Single estimator versus bagging biasvariance decomposition

•Imputing missing values with variants of IterativeImputer

•Using KBinsDiscretizer to discretize continuous features

•Decision Tree Regression

•Multioutput Decision Tree Regression

638sklearnDecision Trees 2343

scikitlearn user guide Release 0213

6383sklearnntree ExtraTreeClassifier

classsklearnntree ExtraTreeClassifier criterion'gini' splitter'random'

maxdepthNone minsamplesplit2

minsamplesleaf1 minweightfractionleaf00

maxfeatures'auto' randomstateNone

maxleafnodesNone minimpuritydecrease00

minimpuritysplitNone classweightNone

An extremely randomized tree classifier

Extratrees differ from classic decision trees in the way they are built When looking for the best split to separate the samples of a node into two groups random splits are drawn for each of the maxfeatures randomly selected features and the best split among those is chosen When maxfeatures is set 1 this amounts to building a totally random decision tree

Warning Extratrees should only be used within ensemble methods

Read more in the User Guide

Parameters

criterion string optional default"gini" The function to measure the quality of a split Sup

ported criteria are "gini" for the Gini impurity and "entropy" for the information gain

splitter string optional default"random" The strategy used to choose the split at each node

Supported strategies are "best" to choose the best split and "random" to choose the best

random split

maxdepth int or None optional defaultNone The maximum depth of the tree If None

then nodes are expanded until all leaves are pure or until all leaves contain less than

minsamplesplit samples

minsamplesplit int float optional default2 The minimum number of samples required to split an internal node

- If int then consider minsamplesplit as the minimum number

- If float then minsamplesplit is a fraction and ceilminsamplesplit

nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions

minsamplesleaf int float optional default1 The minimum number of samples required

to be at a leaf node A split point at any depth will only be considered if it leaves at least

minsamplesleaf training samples in each of the left and right branches This may

have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number

- If float then minsamplesleaf is a fraction and ceilminsamplesleaf

nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the

sum total of weights of all the input samples required to be at a leaf node Samples have equal weight when sampleweight is not provided

maxfeatures int float string or None optional default"auto" The number of features to consider when looking for the best split

- If int then consider maxfeatures features at each split

2344 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- If float then maxfeatures is a fraction and intmaxfeatures nfeatures features are considered at each split
- If “auto” then maxfeaturesqrtnfeatures
- If “sqrt” then maxfeaturesqrtnfeatures
- If “log2” then maxfeatureslog2nfeatures
- If None then maxfeaturesnfeatures

Note the search for a split does not stop until at least one valid partition of the node samples is found even if it requires to effectively inspect more than maxfeatures features

randomstate int RandomState instance or None optional defaultNone If int ran domstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

maxleafnodes int or None optional defaultNone Grow a tree with maxleafnodes in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then unlimited number of leaf nodes

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$Nt \cdot N \cdot \text{impurity} - NtR \cdot Nt \cdot \text{rightimpurity} - NtL \cdot Nt \cdot \text{leftimpurity}$$

whereNis the total number of samples Nt is the number of samples at the current node NtL is the number of samples in the left child and NtR is the number of samples in the right child

NNtNtR andNtL all refer to the weighted sum if sampleweight is passed

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of

minimpuritydecrease in 019 The default value of minimpuritysplit will change from 1e7 to 0 in 023 and it will be removed in 025 Use

minimpuritydecrease instead

classweight dict list of dicts “balanced” or None defaultNone Weights associated with classes in the form classlabel weight If not given all classes are supposed to have weight one For multioutput problems a list of dicts can be provided in the same order as the columns of y

Note that for multioutput including multilabel weights should be defined for each class of every column in its own dict For example for fourclass multilabel classification weights should be 0 1 1 1 0 1 1 5 0 1 1 1 0 1 1 1 instead of 11 25

31 41

The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as nsamples nclasses np

bincounty

For multioutput the weights of each column of y will be multiplied

638sklearntree Decision Trees 2345

scikitlearn user guide Release 0213

Note that these weights will be multiplied with sampleweight passed through the fit method if sampleweight is specified

Attributes

featureimportances Return the feature importances

See also

ExtraTreeRegressor sklearnensembleExtraTreesClassifier

sklearnensembleExtraTreesRegressor

Notes

The default values for the parameters controlling the size of the trees eg maxdepth minsamplesleaf etc lead to fully grown and unpruned trees which can potentially be very large on some data sets To reduce memory consumption the complexity and size of the trees should be controlled by setting those parameter values

References

Rdd99a0224c6e1

Methods

apply self X checkinput Returns the index of the leaf that each sample is predicted as

decisionpath self X checkinput Return the decision path in the tree

fitself X y sampleweight Build a decision tree classifier from the training set X y

getdepth self Returns the depth of the decision tree

getnleaves self Returns the number of leaves of the decision tree

getparams self deep Get parameters for this estimator

predict self X checkinput Predict class or regression value for X

predictlogproba self X Predict class logprobabilities of the input samples X

predictproba self X checkinput Predict class probabilities of the input samples X

score self X y sampleweight Returns the mean accuracy on the given test data and labels

setparams self params Set the parameters of this estimator

init selfcriterion'gini' splitter'random' maxdepthNone minsamplessplit2

minsamplesleaf1 minweightfractionleaf00 maxfeatures'auto'

randomstateNone maxleafnodesNone minimpuritydecrease00

minimpuritysplitNone classweightNone

applyselfXcheckinputTrue

Returns the index of the leaf that each sample is predicted as

New in version 017

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Inter

2346 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

nally it will be converted to dtype np.float32 and if a sparse matrix is provided to a sparse csr matrix

check_input boolean default True Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

X_leaves arraylike shape (n_samples) For each datapoint x in X return the index of the leaf x ends up in Leaves are numbered within 0 self.tree_.n_nodecount

possibly with gaps in the numbering

decision_path self.X_check_input True

Return the decision path in the tree

New in version 0.18

Parameters

X arraylike or sparse matrix shape (n_samples, n_features) The input samples Internally it will be converted to dtype np.float32 and if a sparse matrix is provided to a sparse csr matrix

check_input boolean default True Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

indicator sparse csr array shape (n_samples, n_nodes) Return a node indicator matrix where non zero elements indicates that the samples goes through the nodes

feature importances

Return the feature importances

The importance of a feature is computed as the normalized total reduction of the criterion brought by that feature It is also known as the Gini importance

Returns

feature importances array shape (n_features)

fit_self_Xysample_weight None check_input True X_idxsorted None

Build a decision tree classifier from the training set X y

Parameters

X arraylike or sparse matrix shape (n_samples, n_features) The training input samples Internally it will be converted to dtype np.float32 and if a sparse matrix is provided to a sparse csc matrix

y arraylike shape (n_samples) or (n_samples, n_outputs) The target values class labels as integers or strings

sample_weight arraylike shape (n_samples) or None Sample weights If None then samples are equally weighted Splits that would create child nodes with net zero or negative weight are ignored while searching for a split in each node Splits are also ignored if they would result in any single class carrying a negative weight in either child node

check_input boolean default True Allow to bypass several input checking Don't use this parameter unless you know what you do

X_idxsorted arraylike shape (n_samples, n_features) optional The indexes of the sorted training input samples If many trees are grown on the same dataset this allows the

638 sklearn.tree Decision Trees 2347

scikitlearn user guide Release 0213

ordering to be cached between trees If None the data will be sorted here Don't use this parameter unless you know what to do

Returns

self object

getdepth self

Returns the depth of the decision tree

The depth of a tree is the maximum distance between the root and any leaf

getnleaves self

Returns the number of leaves of the decision tree

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXcheckinputTrue

Predict class or regression value for X

For a classification model the predicted class for each sample in X is returned For a regression model

the predicted value based on X is returned

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Inter

nally it will be converted to dtype npfloat32 and if a sparse matrix is provided to

a sparsecsrmatrix

checkinput boolean defaultTrue Allow to bypass several input checking Don't use

this parameter unless you know what you do

Returns

yarray of shape nsamples or nsamples noutputs The predicted classes or the

predict values

predictlogproba selfX

Predict class logprobabilities of the input samples X

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Inter

nally it will be converted to dtype npfloat32 and if a sparse matrix is provided to

a sparsecsrmatrix

Returns

parray of shape nsamples nclasses or a list of noutputs such arrays if noutputs

1 The class logprobabilities of the input samples The order of the classes corresponds

to that in the attribute classes

predictproba selfXcheckinputTrue

Predict class probabilities of the input samples X

The predicted class probability is the fraction of samples of the same class in a leaf

2348 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Parameters

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsrmatrix

checkinput bool Run checkarray on X

Returns

array of shape nsamples nclasses or a list of noutputs such arrays if noutputs

1 The class probabilities of the input samples The order of the classes corresponds to that in the attribute classes

scoreselfXysampleweightNone

Returns the mean accuracy on the given test data and labels

In multilabel classification this is the subset accuracy which is a harsh metric since you require for each sample that each label set be correctly predicted

Parameters

Xarraylike shape nsamples nfeatures Test samples

yarraylike shape nsamples or nsamples noutputs True labels for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float Mean accuracy of selfpredictX wrt y

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

6384sklearn.tree.ExtraTreeRegressor

class sklearn.tree.ExtraTreeRegressor criterion'mse' splitter'random' maxdepthNone

minsamplesplit2 minsamplesleaf1

minweightfractionleaf00 maxfeatures'auto'

randomstateNone minimpuritydecrease00

minimpuritysplitNone maxleafnodesNone

An extremely randomized tree regressor

Extratrees differ from classic decision trees in the way they are built When looking for the best split to separate the samples of a node into two groups random splits are drawn for each of the maxfeatures randomly selected features and the best split among those is chosen When maxfeatures is set 1 this amounts to building a totally random decision tree

Warning Extratrees should only be used within ensemble methods

638sklearn.tree.Decision Trees 2349

scikitlearn user guide Release 0213

Read more in the User Guide

Parameters

criterion string optional default"mse" The function to measure the quality of a split Supported criteria are "mse" for the mean squared error which is equal to variance reduction as feature selection criterion and "mae" for the mean absolute error

New in version 018 Mean Absolute Error MAE criterion

splitter string optional default"random" The strategy used to choose the split at each node

Supported strategies are "best" to choose the best split and "random" to choose the best

random split

maxdepth int or None optional defaultNone The maximum depth of the tree If None

then nodes are expanded until all leaves are pure or until all leaves contain less than

minsamplesplit samples

minsamplesplit int float optional default2 The minimum number of samples required

to split an internal node

- If int then consider minsamplesplit as the minimum number
- If float then minsamplesplit is a fraction and ceilminsamplesplit

nsamples are the minimum number of samples for each split

Changed in version 018 Added float values for fractions

minsamplesleaf int float optional default1 The minimum number of samples required

to be at a leaf node A split point at any depth will only be considered if it leaves at least

minsamplesleaf training samples in each of the left and right branches This may

have the effect of smoothing the model especially in regression

- If int then consider minsamplesleaf as the minimum number
- If float then minsamplesleaf is a fraction and ceilminsamplesleaf

nsamples are the minimum number of samples for each node

Changed in version 018 Added float values for fractions

minweightfractionleaf float optional default0 The minimum weighted fraction of the sum total of weights of all the input samples required to be at a leaf node Samples have equal weight when sampleweight is not provided

maxfeatures int float string or None optional default"auto" The number of features to

consider when looking for the best split

- If int then consider maxfeatures features at each split
- If float then maxfeatures is a fraction and intmaxfeatures

nfeatures features are considered at each split

- If "auto" then maxfeaturesnfeatures
- If "sqrt" then maxfeaturesqrtnfeatures
- If "log2" then maxfeatureslog2nfeatures
- If None then maxfeaturesnfeatures

Note the search for a split does not stop until at least one valid partition of the node samples

is found even if it requires to effectively inspect more than maxfeatures features

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

2350 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

minimpuritydecrease float optional default0 A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following

$$Nt - N \text{ impurity} - NtR - Nt \text{ rightimpurity}$$

$$NtL - Nt \text{ leftimpurity}$$

whereNis the total number of samples Nt is the number of samples at the current node

NtL is the number of samples in the left child and NtR is the number of samples in the right child

NNtNtR andNtL all refer to the weighted sum if sampleweight is passed

New in version 019

minimpuritysplit float default1e7 Threshold for early stopping in tree growth A node

will split if its impurity is above the threshold otherwise it is a leaf

Deprecated since version 019 minimpuritysplit has been deprecated in favor of

minimpuritydecrease in 019 The default value of minimpuritysplit

will change from 1e7 to 0 in 023 and it will be removed in 025 Use

minimpuritydecrease instead

maxleafnodes int or None optional defaultNone Grow a tree with maxleafnodes

in bestfirst fashion Best nodes are defined as relative reduction in impurity If None then

unlimited number of leaf nodes

Attributes

featureimportances Return the feature importances

See also

ExtraTreeClassifier sklearnensembleExtraTreesClassifier

sklearnensembleExtraTreesRegressor

Notes

The default values for the parameters controlling the size of the trees eg maxdepth

minsamplesleaf etc lead to fully grown and unpruned trees which can potentially be very large on

some data sets To reduce memory consumption the complexity and size of the trees should be controlled by

setting those parameter values

References

R4939d63d5a491

Methods

638sklearnrtree Decision Trees 2351

scikitlearn user guide Release 0213

apply self X checkinput Returns the index of the leaf that each sample is predicted as

decisionpath self X checkinput Return the decision path in the tree

fitself X y sampleweight Build a decision tree regressor from the training set X y

getdepth self Returns the depth of the decision tree

getnleaves self Returns the number of leaves of the decision tree

getparams self deep Get parameters for this estimator

predict self X checkinput Predict class or regression value for X

score self X y sampleweight Returns the coefficient of determination R2 of the prediction

setparams self params Set the parameters of this estimator

init selfcriterion'mse' splitter'random' maxdepthNone minsamplesplit2

minsamplesleaf1 minweightfractionleaf00 maxfeatures'auto'

randomstateNone minimpuritydecrease00 minimpuritysplitNone

maxleafnodesNone

applyselfXcheckinputTrue

Returns the index of the leaf that each sample is predicted as

New in version 017

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

Xleaves arraylike shape nsamples For each datapoint x in X return the index of the leaf x ends up in Leaves are numbered within 0 selftree nodecount possibly with gaps in the numbering

decisionpath selfXcheckinputTrue

Return the decision path in the tree

New in version 018

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The input samples Internally it will be converted to dtype npfloat32 and if a sparse matrix is provided to a sparsecsr matrix

checkinput boolean defaultTrue Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

indicator sparse csr array shape nsamples nnodes Return a node indicator matrix where non zero elements indicates that the samples goes through the nodes feature importances

Return the feature importances

2352 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

The importance of a feature is computed as the normalized total reduction of the criterion brought by that feature It is also known as the Gini importance

Returns

featureimportances array shape nfeatures

fitselfXysampleweightNone checkinputTrue XidxsortedNone

Build a decision tree regressor from the training set X y

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures The training input samples

Internally it will be converted to dtypefloat32 and if a sparse matrix is provided to a sparsecscmatrix

yarraylike shape nsamples or nsamples noutputs The target values real num

bers Useddtypefloat64 andorderC for maximum efficiency

sampleweight arraylike shape nsamples or None Sample weights If None then

samples are equally weighted Splits that would create child nodes with net zero or nega

tive weight are ignored while searching for a split in each node

checkinput boolean defaultTrue Allow to bypass several input checking Don't use

this parameter unless you know what you do

Xidxsorted arraylike shape nsamples nfeatures optional The indexes of the

sorted training input samples If many tree are grown on the same dataset this allows the

ordering to be cached between trees If None the data will be sorted here Don't use this

parameter unless you know what to do

Returns

self object

getdepth self

Returns the depth of the decision tree

The depth of a tree is the maximum distance between the root and any leaf

getnleaves self

Returns the number of leaves of the decision tree

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

predictselfXcheckinputTrue

Predict class or regression value for X

For a classification model the predicted class for each sample in X is returned For a regression model

the predicted value based on X is returned

Parameters

638sklearnntree Decision Trees 2353

scikitlearn user guide Release 0213

Xarraylike or sparse matrix of shape nsamples nfeatures The input samples Internally it will be converted to dtype np.float32 and if a sparse matrix is provided to a sparse csr matrix

checkinput boolean default True Allow to bypass several input checking Don't use this parameter unless you know what you do

Returns

yarray of shape nsamples or nsamples noutputs The predicted classes or the predict values

score self.Xysampleweight None

Returns the coefficient of determination R2 of the prediction

The coefficient R2 is defined as $1 - \frac{u}{v}$ where u is the residual sum of squares $y_{true} - y_{pred}$

$2sum$ and v is the total sum of squares $y_{true} - y_{true\text{mean}}$ $2sum$ The best possible score is 1.0 and it can be negative because the model can be arbitrarily worse A constant model that always predicts the expected value of y disregarding the input features would get a R2 score of 0.0

Parameters

Xarraylike shape nsamples nfeatures Test samples For some estimators this may

be a precomputed kernel matrix instead shape nsamples nsamplesfitted where nsamplesfitted is the number of samples used in the fitting for the estimator

yarraylike shape nsamples or nsamples noutputs True values for X

sampleweight arraylike shape nsamples optional Sample weights

Returns

score float R2 of self.predict(X) wrt y

Notes

The R2 score used when calling score on a regressor will use multioutputuniformaverage

from version 0.23 to keep consistent with metrics.r2score This will influence the score

method of all the multioutput regressors except for multioutput.MultiOutputRegressor

To specify the default value manually and avoid the warning please either call metrics.r2score

directly or make a custom scorer with metrics.make_scorer the builtin scorer r2 uses

multioutputuniformaverage

set_params self.params

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form component.parameter so that it's possible to update each component

of a nested object

Returns

self

tree.export_graphviz decisiontree Export a decision tree in DOT format

tree.plot_tree decisiontree maxdepth Plot a decision tree

tree.export_text decisiontree Build a text report showing the rules of a decision tree

2354 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

6385sklearn.tree.export_graphviz

sklearn.tree.export_graphviz decisiontree outfile=None maxdepth=None feature_names=None class_names=None label='all' filled=False leaves_parallel=False impurity=True node_ids=False proportion=False rotate=False rounded=False special_characters=False precision=3

Export a decision tree in DOT format

This function generates a GraphViz representation of the decision tree which is then written into outfile

Once exported graphical renderings can be generated using for example

- dot Tps treedot o treeps PostScript format
- dot Tpng treedot o treepng PNG format

The sample counts that are shown are weighted with any sample weights that might be present

Read more in the User Guide

Parameters

decisiontree decision tree classifier The decision tree to be exported to GraphViz

outfile file object or string optional default=None Handle or name of the output file If None the result is returned as a string

Changed in version 020 Default of outfile changed from “treedot” to None

maxdepth int optional default=None The maximum depth of the representation If None the tree is fully generated

feature_names list of strings optional default=None Names of each of the features

class_names list of strings bool or None optional default=None Names of each of the target classes in ascending numerical order Only relevant for classification and not supported for multioutput

If True shows a symbolic representation of the class name

label 'all' 'root' 'none' optional default='all' Whether to show informative labels for impurity etc Options include 'all' to show at every node 'root' to show only at the top root node or 'none' to not show at any node

filled bool optional default=False When set to True paint nodes to indicate majority class for classification extremity of values for regression or purity of node for multioutput

leaves_parallel bool optional default=False When set to True draw all leaf nodes at the bottom of the tree

impurity bool optional default=True When set to True show the impurity at each node

node_ids bool optional default=False When set to True show the ID number on each node

proportion bool optional default=False When set to True change the display of 'values' and/or 'samples' to be proportions and percentages respectively

rotate bool optional default=False When set to True orient tree left to right rather than topdown

rounded bool optional default=False When set to True draw node boxes with rounded corners and use Helvetica fonts instead of TimesRoman

special_characters bool optional default=False When set to False ignore special characters for PostScript compatibility

638sklearn.tree Decision Trees 2355

scikitlearn user guide Release 0213

precision int optional default3 Number of digits of precision for floating point in the values of impurity threshold and value attributes of each node

Returns
dotdata string String representation of the input tree in GraphViz dot format Only returned if outfile is None

New in version 018

Examples
from sklearn.datasets import load_iris
from sklearn import tree
clf = tree.DecisionTreeClassifier

iris = load_iris
clf = clf.fit(iris.data, iris.target)
tree.export_graphviz(clf,

digraph Tree
6386sklearn.tree.plot_tree
sklearn.tree.plot_tree(decision_tree, max_depth=None, feature_names=None, class_names=None, label='all', filled=False, impurity=True, node_ids=False, proportion=False, rotate=False, rounded=False, precision=3, ax=None, font_size=None)

Plot a decision tree
The sample counts that are shown are weighted with any sample weights that might be present This function requires matplotlib and works best with matplotlib 1.5
The visualization is fit automatically to the size of the axis Use the figsize or dpi arguments of plt.figure to control the size of the rendering

Read more in the User Guide

New in version 021

Parameters
decision_tree decision tree regressor or classifier The decision tree to be exported to GraphViz
max_depth int optional defaultNone The maximum depth of the representation If None the tree is fully generated
feature_names list of strings optional defaultNone Names of each of the features
class_names list of strings bool or None optional defaultNone Names of each of the target classes in ascending numerical order Only relevant for classification and not supported for multioutput If True shows a symbolic representation of the class name
label 'all' 'root' 'none' optional default'all' Whether to show informative labels for impurity etc Options include 'all' to show at every node 'root' to show only at the top root node or 'none' to not show at any node

2356 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

filled bool optional defaultFalse When set to True paint nodes to indicate majority class for classification extremity of values for regression or purity of node for multioutput

impurity bool optional defaultTrue When set to True show the impurity at each node

nodeids bool optional defaultFalse When set to True show the ID number on each node

proportion bool optional defaultFalse When set to True change the display of 'values' andor 'samples' to be proportions and percentages respectively

rotate bool optional defaultFalse When set to True orient tree left to right rather than topdown

rounded bool optional defaultFalse When set to True draw node boxes with rounded corners and use Helvetica fonts instead of TimesRoman

precision int optional default3 Number of digits of precision for floating point in the values of impurity threshold and value attributes of each node

axmatplotlib axis optional defaultNone Axes to plot to If None use current axis Any previous content is cleared

fontsize int optional defaultNone Size of text font If None determined automatically to fit figure

Returns
annotations list of artists List containing the artists for the annotation boxes making up the tree

Examples
from sklearn.datasets import loadiris
from sklearn import tree
clf = tree.DecisionTreeClassifier(random_state=0)
iris = loadiris
clf = clf.fit(iris.data, iris.target)
tree.plot_tree(clf)
Text2515345217X3 08

Examples using sklearn.tree.plot_tree
•Plot the decision surface of a decision tree on the iris dataset

6387sklearn.tree.export_text
sklearn.tree.export_text(decision_tree, feature_names=None, max_depth=10, spacing=3, decimals=2, show_weights=False)

Build a text report showing the rules of a decision tree
Note that backwards compatibility may not be supported
Parameters

638sklearn.tree Decision Trees 2357

scikitlearn user guide Release 0213

decisiontree object The decision tree estimator to be exported It can be an instance of DecisionTreeClassifier or DecisionTreeRegressor
featurenames list optional defaultNone A list of length nfeatures containing the feature names If None generic names will be used "feature0" "feature1"
maxdepth int optional default10 Only the first maxdepth levels of the tree are exported Truncated branches will be marked with " "
spacing int optional default3 Number of spaces between edges The higher it is the wider the result

decimals int optional default2 Number of decimal digits to display
showweights bool optional defaultFalse If true the classification weights will be exported on each leaf The classification weights are the number of samples each class
Returns

report string Text summary of all the rules in the decision tree

Examples

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree.export import export_text

iris = load_iris()
X, y = iris.data, iris.target
decisiontree = DecisionTreeClassifier(random_state=0, max_depth=2)
decisiontree.fit(X, y)
r = export_text(decisiontree, feature_names=iris.feature_names)
print(r)

petal width cm 0.80
class 0
petal width cm 0.80
petal width cm 1.75
class 1
petal width cm 1.75
class 2
```

639sklearnutils Utilities

The sklearnutils module includes various utilities
Developer guide See the Utilities for Developers page for further details
utils.arrayfuncs.choleskydelete L
goout
utils.arrayfuncs.minpos Find the minimum value of an array over positive values
utils.asfloatarray X copy forceallfinite Converts an arraylike to an array of floats
utils.assertallfinite X allownan Throw a ValueError if X contains NaN or infinity
utils.checkXy X y acceptspare Input validation for standard estimators
Continued on next page

| | | | | |
|------------------------------|--|--|--|--|
| utilscheckarray | array | acceptsparse | Input validation on an array list sparse matrix or similar | |
| utilscheckscalar | x | name | targettype | Validate scalar parameters type and value |
| utilscheckconsistentlength | arrays | Check that all arrays have consistent first dimensions | | |
| utilscheckrandomstate | seed | Turn seed into a nprandomRandomState instance | | |
| utilsclassweight | | | | |
| computeclassestweight | | Estimate class weights for unbalanced datasets | | |
| utilsclassweight | | | | |
| computesampleweight | | Estimate sample weights by class for unbalanced datasets | | |
| utilsdeprecated | extra | Decorator to mark a function or class as deprecated | | |
| utilssestimatorchecks | | | | |
| checksestimator | Estimator | Check if estimator adheres to scikitlearn conventions | | |
| utilsextmathsafesparse | dot | a | b | Dot product that handle the sparse matrix case correctly |
| utilsextmathrandomizedrange | finder | A | | Computes an orthonormal matrix whose range approximates the range of A |
| utilsextmathrandomizedsvd | M | | | |
| ncomponents | Computes a truncated randomized SVD | | | |
| utilsextmathfastlogdet | A | Compute logdetA for A symmetric | | |
| utilsextmathdensity | w | kwargs | Compute density of a sparse vector | |
| utilsextmathweightedmode | a | w | axis | Returns an array of the weighted modal most common value in a |
| utilsgenevenslices | n | npacks | nsamples | Generator to create npacks slices going up to n |
| utilsgraphsinglesource | shortestpathlength | | | Return the shortest path length from source to all reachable nodes |
| utilsgraphshortestpath | | | | |
| graphshortestpath | Perform a shortestpath graph search on a positive directed or undirected graph | | | |
| utilsindexable | iterables | Make arrays indexable for crossvalidation | | |
| utilsmetaestimators | | | | |
| ifdelegatehasmethod | | Create a decorator for methods that are delegated to a sub estimator | | |
| utilsmulticlassdtypeoftarget | y | Determine the type of data indicated by the target | | |
| utilsmulticlassismultilabel | y | Check if y is in a multilabel format | | |
| utilsmulticlassunique | labels | ys | Extract an ordered array of unique labels | |
| utilsmurmurhash32 | key | at seed | | Compute the 32bit murmurhash3 of key at seed |
| utilssample | arrays | options | Resample arrays or sparse matrices in a consistent way | |
| utilssafeindexing | X | indices | Return items or rows from X using indices | |
| utilssafemask | X | mask | Return a mask which is safe to use on X | |
| utilssafesqr | X | copy | Element wise squaring of arraylikes and sparse matrices | |
| utilsshuffle | arrays | options | Shuffle arrays or sparse matrices in a consistent way | |
| utilssparsefuncs | | | | |
| incrmeanvarianceaxis | X | Compute incremental mean and variance along an axis on a CSR (Compressed Sparse Row) or CSC matrix | | |
| utilssparsefuncs | | | | |
| inplacecolumnscale | X | scaleInplace column scaling of a CSCCSR matrix | | |
| utilssparsefuncsinplacero | wscale | X | | |
| scaleInplace | row scaling of a CSR (Compressed Sparse Row) or CSC matrix | | | |
| utilssparsefuncsinplaceswap | row | X | m | |
| nSwaps | two rows of a CSCCSR matrix | inplace | | |
| utilssparsefuncs | | | | |
| inplaceswapcolumn | X | m | nSwaps two columns of a CSCCSR matrix | inplace |
| utilssparsefuncsmeanvariance | axis | X | | |
| axis | Compute mean and variance along an axis on a CSR (Compressed Sparse Row) or CSC matrix | | | |
| utilssparsefuncs | | | | |
| inplacecsrcolumnscale | X | Inplace column scaling of a CSR (Compressed Sparse Row) matrix | | |

Continued on next page

639sklearnutils Utilities 2359

Continued on next page

scikitlearn user guide Release 0213

Table 6288 – continued from previous page

utilssparsfuncsfast

inplacecsrrownormalizel1 Inplace row normalize using the l1 norm

utilssparsfuncsfast

inplacecsrrownormalizel2 Inplace row normalize using the l2 norm

utilsrandomsamplewithoutreplacement Sample integers without replacement

utilvalidationcheckisfitted estimator

Perform isfitted validation for estimator

utilvalidationcheckmemory memory Check that memory is joblibMemorylike

utilvalidationchecksymmetric array

Make sure that array is 2D square and symmetric

utilvalidationcolumnor1d y warn Ravel column or 1d numpy array else raises an error

utilvalidationhasfitparameter Checks whether the estimator’s fit method supports the given parameter

utiltestingassertin member container

msgJust like selfassertTruea in b but with a nicer default

message

utiltestingassertnotin member con

tainerJust like selfassertTruea not in b but with a nicer default

message

utiltestingassertraisemessage Helper function to test the message raised in an exception

utiltestingallemestimators Get a list of all estimators from sklearn

6391sklearnutils arrayfuncscholeskydelete

6392sklearnutilsarrayfuncs minpos

sklearnutilsarrayfuncs minpos

Find the minimum value of an array over positive values

Returns a huge value if none of the values are positive

6393sklearnutils asfloatarray

sklearnutils asfloatarray XcopyTrue forceallfiniteTrue

Converts an arraylike to an array of floats

The new dtype will be npfloat32 or npfloat64 depending on the original type The function can create a copy

or modify the argument depending on the argument copy

Parameters

Xarraylike sparse matrix

copy bool optional If True a copy of X will be created If False a copy may still be returned

if X’s dtype is not a floating point type

forceallfinite boolean or ‘allownan’ defaultTrue Whether to raise an error on npinf

and npnan in X The possibilities are

- True Force all values of X to be finite
- False accept both npinf and npnan in X
- ‘allownan’ accept only npnan values in X Values cannot be infinite

New in version 020 forceallfinite accepts the string allownan

Returns

2360 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

XT array sparse matrix An array of type npfloat

6394sklearnutils assertallfinite

sklearnutils assertallfinite XallownanFalse

Throw a ValueError if X contains NaN or infinity

Parameters

Xarray or sparse matrix

allownan bool

6395sklearnutils checkXy

sklearnutils checkXy XyacceptsparseFalse acceptlargesparsedTrue dtype'numeric'

orderNone copyFalse forceallfiniteTrue ensure2dTrue

allowndFalse multioutputFalse ensureminsamples1 en

sureminfeatures1 ynumericFalse warnondtypeNone estima

torNone

Input validation for standard estimators

Checks X and y for consistent length enforces X to be 2D and y 1D By default X is checked to be nonempty and containing only finite values Standard input checks are also applied to y such as checking that y does not have npnan or npinf targets For multilabel y set multioutputTrue to allow 2D and sparse y If the dtype of X is object attempt converting to float raising on failure

Parameters

Xndarray list or sparse matrix Input data

yndarray list or sparse matrix Labels

acceptsparse string boolean or list of string defaultFalse Strings representing allowed

sparse matrix formats such as 'csc' 'csr' etc If the input is sparse but not in the allowed format it will be converted to the first listed format True allows the input to be any format False means that a sparse matrix input will raise an error

acceptlargesparsed bool defaultTrue If a CSR (Compressed Sparse Row) CSC COO or BSR sparse matrix is supplied and accepted by acceptsparse acceptlargesparsed will cause it to be accepted only if its indices are stored with a 32bit dtype

New in version 020

dtype string type list of types or None default"numeric" Data type of result If None the dtype of the input is preserved If "numeric" dtype is preserved unless arraydtype is object If dtype is a list of types conversion on the first type is only performed if the dtype of the input is not in the list

order 'F' 'C' or None defaultNone Whether an array will be forced to be fortran or cstyle

copy boolean defaultFalse Whether a forced copy will be triggered If copyFalse a copy might be triggered by a conversion

forceallfinite boolean or 'allownan' defaultTrue Whether to raise an error on npinf and npnan in X This parameter does not influence whether y can have npinf or npnan values The possibilities are

- True Force all values of X to be finite

639sklearnutils Utilities 2361

scikitlearn user guide Release 0213

- False accept both npinf and npnan in X
- ‘allownan’ accept only npnan values in X Values cannot be infinite

New in version 020 forceallfinite accepts the string allownan

ensure2d boolean defaultTrue Whether to raise a value error if X is not 2D

allownd boolean defaultFalse Whether to allow Xndim 2

multioutput boolean defaultFalse Whether to allow 2D y array or sparse matrix

If false y will be validated as a vector y cannot have npnan or npinf values if

multioutputTrue

ensureminsamples int default1 Make sure that X has a minimum number of samples in its first axis rows for a 2D array

ensureminfeatures int default1 Make sure that the 2D array has some minimum number of features columns The default value of 1 rejects empty datasets This check is only enforced when X has effectively 2 dimensions or is originally 1D and ensure2d is True Setting to 0 disables this check

ynumeric boolean defaultFalse Whether to ensure that y has a numeric type If dtype of

y is object it is converted to float64 Should only be used for regression algorithms

warnondtype boolean or None optional defaultNone Raise DataConversionWarning if the dtype of the input data structure does not match the requested dtype causing a memory copy

Deprecated since version 021 warnondtype is deprecated in version 021 and will be removed in 023

estimator str or estimator instance defaultNone If passed include the name of the estimator in warning messages

Returns

Xconverted object The converted and validated X

yconverted object The converted and validated y

6396sklearnutils checkarray

sklearnutils checkarray array acceptsparseFalse acceptlargesparseTrue

dtype‘numeric’ orderNone copyFalse forceallfiniteTrue

ensure2dTrue allowndFalse ensureminsamples1 en

sureminfeatures1 warnondtypeNone estimatorNone

Input validation on an array list sparse matrix or similar

By default the input is checked to be a nonempty 2D array containing only finite values If the dtype of the array is object attempt converting to float raising on failure

Parameters

array object Input object to check convert

acceptsparse string boolean or listtuple of strings defaultFalse Strings representing

allowed sparse matrix formats such as ‘csc’ ‘csr’ etc If the input is sparse but not in the

allowed format it will be converted to the first listed format True allows the input to be any format False means that a sparse matrix input will raise an error

2362 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

acceptlargesparsed bool defaultTrue If a CSR (Compressed Sparse Row) CSC COO or BSR sparse matrix is supplied and accepted by acceptsparsed acceptlargesparsedFalse will cause it to be accepted only if its indices are stored with a 32bit dtype

New in version 020

dtype string type list of types or None default"numeric" Data type of result If None the dtype of the input is preserved If "numeric" dtype is preserved unless arraydtype is object If dtype is a list of types conversion on the first type is only performed if the dtype of the input is not in the list

order 'F' 'C' or None defaultNone Whether an array will be forced to be fortran or cstyle When order is None default then if copyFalse nothing is ensured about the memory layout of the output array otherwise copyTrue the memory layout of the returned array is kept as close as possible to the original array

copy boolean defaultFalse Whether a forced copy will be triggered If copyFalse a copy might be triggered by a conversion

forceallfinite boolean or 'allownan' defaultTrue Whether to raise an error on npinf and npnan in array The possibilities are

- True Force all values of array to be finite
- False accept both npinf and npnan in array
- 'allownan' accept only npnan values in array Values cannot be infinite

For object dtypes data only npnan is checked and not npinf

New in version 020 forceallfinite accepts the string allownan

ensure2d boolean defaultTrue Whether to raise a value error if array is not 2D

allownd boolean defaultFalse Whether to allow arrayndim 2

ensureminsamples int default1 Make sure that the array has a minimum number of samples in its first axis rows for a 2D array Setting to 0 disables this check

ensureminfeatures int default1 Make sure that the 2D array has some minimum number of features columns The default value of 1 rejects empty datasets This check is only enforced when the input data has effectively 2 dimensions or is originally 1D and ensure2d is True Setting to 0 disables this check

warnondtype boolean or None optional defaultNone Raise DataConversionWarning if the dtype of the input data structure does not match the requested dtype causing a memory copy

Deprecated since version 021 warnondtype is deprecated in version 021 and will be removed in 023

estimator str or estimator instance defaultNone If passed include the name of the estimator in warning messages

Returns

arrayconverted object The converted and validated array

6397sklearnutils checkscalar

sklearnutils checkscalar xname targettype minvalNone maxvalNone

Validate scalar parameters type and value

639sklearnutils Utilities 2363

Parameters

xobject The scalar parameter to validate
name str The name of the parameter to be printed in error messages
targettype type or tuple Acceptable data types for the parameter
minval float or int optional defaultNone The minimum valid value the parameter can take If None default it is implied that the parameter does not have a lower bound
maxval float or int optional defaultNone The maximum valid value the parameter can take If None default it is implied that the parameter does not have an upper bound

Raises

TypeError If the parameter’s type does not match the desired type

ValueError If the parameter’s value violates the given bounds

6398sklearnutils checkconsistentlength
sklearnutils checkconsistentlength arrays

Check that all arrays have consistent first dimensions

Checks whether all objects in arrays have the same shape or length

Parameters

arrays list or tuple of input objects Objects that will be checked for consistent length

6399sklearnutils checkrandomstate

sklearnutils checkrandomstate seed

Turn seed into a nprandomRandomState instance

Parameters

seed None int instance of RandomState If seed is None return the RandomState singleton

used by nprandom If seed is an int return a new RandomState instance seeded with seed

If seed is already a RandomState instance return it Otherwise raise ValueError

Examples using sklearnutilscheckrandomstate

- Isotonic Regression
 - Face completion with a multioutput estimators
 - Empirical evaluation of the impact of kmeans initialization
 - MNIST classffication using multinomial logistic L1
 - Manifold Learning methods on a severed sphere
 - Scaling the regularization parameter for SVCs
- 2364 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
63910sklearnutilsclassweight computeclassweight
sklearnutilsclassweight computeclassweight classweight classes y
Estimate class weights for unbalanced datasets
Parameters
classweight dict 'balanced' or None If 'balanced' class weights will be given by
nsamples nclasses npbincounty If a dictionary is given keys
are classes and values are corresponding class weights If None is given the class weights
will be uniform
classes ndarray Array of the classes occurring in the data as given by npuniqueyorg
withyorg the original class labels
yarraylike shape nsamples Array of original class labels per sample
Returns
classweightvect ndarray shape nclasses Array with classweightvecti the weight for
ith class
References
The "balanced" heuristic is inspired by Logistic Regression in Rare Events Data King Zen 2001
63911sklearnutilsclassweight computesampleweight
sklearnutilsclassweight computesampleweight classweight yindicesNone
Estimate sample weights by class for unbalanced datasets
Parameters
classweight dict list of dicts "balanced" or None optional Weights associated with classes
in the form classlabel weight If not given all classes are supposed to have
weight one For multioutput problems a list of dicts can be provided in the same order as
the columns of y
Note that for multioutput including multilabel weights should be defined for each class of
every column in its own dict For example for fourclass multilabel classification weights
should be 0 1 1 1 0 1 1 5 0 1 1 1 0 1 1 1 instead of 11 25
31 41
The "balanced" mode uses the values of y to automatically adjust weights inversely pro
portional to class frequencies in the input data nsamples nclasses np
bincounty
For multioutput the weights of each column of y will be multiplied
yarraylike shape nsamples or nsamples noutputs Array of original class labels per
sample
indices arraylike shape nsubsample or None Array of indices to be used in a subsample
Can be of length less than nsamples in the case of a subsample or equal to nsamples in
the case of a bootstrap subsample with repeated indices If None the sample weight will
be calculated over the full sample Only "balanced" is supported for classweight if this is
provided
Returns
639sklearnutils Utilities 2365

scikitlearn user guide Release 0213

sampleweightvect ndarray shape nsamples Array with sample weights as applied to the original y

63912sklearnutils deprecated

sklearnutils deprecated extra''

Decorator to mark a function or class as deprecated

Issue a warning when the function is calledthe class is instantiated and adds a warning to the docstring

The optional extra argument will be appended to the deprecation message and the docstring Note to use this with the default value for extra put in an empty of parentheses

from sklearnutils import deprecated

deprecated

sklearnutilsdeprecationdeprecated object at

deprecated

def somefunction pass

Parameters

extra string to be added to the deprecation messages

63913sklearnutilsestimatorchecks checkestimator

sklearnutilsestimatorchecks checkestimator Estimator

Check if estimator adheres to scikitlearn conventions

This estimator will run an extensive testsuite for input validation shapes etc Additional tests for classifiers regressors clustering or transformers will be run if the Estimator class inherits from the corresponding mixin from sklearnbase

This test can be applied to classes or instances Classes currently have some additional tests that related to construction while passing instances allows the testing of multiple options

Parameters

estimator estimator object or class Estimator to check Estimator is a class object or instance

63914sklearnutilsextmath safesparsedot

sklearnutilsextmath safesparsedot abdenseoutputFalse

Dot product that handle the sparse matrix case correctly

Uses BLAS GEMM as replacement for numpydot where possible to avoid unnecessary copies

Parameters

aarray or sparse matrix

barray or sparse matrix

denseoutput boolean default False When False either aorbbbeing sparse will yield sparse output When True output will always be an array

Returns

2366 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

dotproduct array or sparse matrix sparse if aorbis sparse and denseoutputFalse

63915sklearnutilsextmath randomizedrangefinder

sklearnutilsextmath randomizedrangefinder A size niter

poweriterationnormalizer'auto'

randomstateNone

Computes an orthonormal matrix whose range approximates the range of A

Parameters

A2D array The input data matrix

size integer Size of the return array

niter integer Number of power iterations used to stabilize the result

poweriterationnormalizer 'auto' default 'QR' 'LU' 'none' Whether the power iter

ations are normalized with stepbystep QR factorization the slowest but most accurate

'none' the fastest but numerically unstable when niter is large eg typically 5 or

larger or 'LU' factorization numerically stable but can lose slightly in accuracy The

'auto' mode applies no normalization if niter > 2 and switches to LU otherwise

New in version 0.18

randomstate int RandomState instance or None optional defaultNone The seed of the

pseudo random number generator to use when shuffling the data If int randomstate is

the seed used by the random number generator If RandomState instance randomstate is

the random number generator If None the random number generator is the RandomState

instance used by np.random

Returns

Q2D array A size x size projection matrix the range of which approximates well the range

of the input matrix A

Notes

Follows Algorithm 4.3 of Finding structure with randomness Stochastic algorithms for constructing approxi

mate matrix decompositions Halko et al 2009 arXiv:0909.4061v1 [math.NA] <https://arxiv.org/pdf/0909.4061.pdf>

An implementation of a randomized algorithm for principal component analysis A Szlam et al 2014

63916sklearnutilsextmath randomizedsvd

sklearnutilsextmath randomizedsvd M ncomponents noversamples10 niter'auto'

poweriterationnormalizer'auto' transpose'auto'

flipsignTrue randomstate0

Computes a truncated randomized SVD

Parameters

M ndarray or sparse matrix Matrix to decompose

ncomponents int Number of singular values and vectors to extract

noversamples int default is 10 Additional number of random vectors to sample the range

of M so as to ensure proper conditioning The total number of random vectors used to find

639sklearnutils Utilities 2367

scikitlearn user guide Release 0213

the range of M is ncomponents noversamples Smaller number can improve speed but can negatively impact the quality of approximation of singular vectors and singular values
niter int or 'auto' default is 'auto' Number of power iterations It can be used to deal with very noisy problems When 'auto' it is set to 4 unless ncomponents is small
1 minXshape niter in which case is set to 7 This improves precision with few components

Changed in version 018

poweriterationnormalizer 'auto' default 'QR' 'LU' 'none' Whether the power iterations are normalized with stepbystep QR factorization the slowest but most accurate 'none' the fastest but numerically unstable when niter is large eg typically 5 or larger or 'LU' factorization numerically stable but can lose slightly in accuracy The 'auto' mode applies no normalization if niter 2 and switches to LU otherwise

New in version 018

transpose True False or 'auto' default Whether the algorithm should be applied to MT instead of M The result should approximately be the same The 'auto' mode will trigger the transposition if Mshape1 Mshape0 since this implementation of randomized SVD tend to be a little faster in that case

Changed in version 018

flipsign boolean True by default The output of a singular value decomposition is only unique up to a permutation of the signs of the singular vectors If flipsign is set to True the sign ambiguity is resolved by making the largest loadings for each component in the left singular vectors positive

randomstate int RandomState instance or None optional defaultNone The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

Notes

This algorithm finds a usually very good approximate truncated singular value decomposition using random ization to speed up the computations It is particularly fast on large matrices on which you wish to extract only a small number of components In order to obtain further speed up niter can be set 2 at the cost of loss of precision

References

- Finding structure with randomness Stochastic algorithms for constructing approximate matrix decompositions Halko et al 2009 <https://arxiv.org/abs/0909.4061>
 - A randomized algorithm for the decomposition of matrices PerGunnar Martinsson Vladimir Rokhlin and Mark Tygert
 - An implementation of a randomized algorithm for principal component analysis A Szlam et al 2014
- 2368 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

63917sklearnutilsextmath fastlogdet
sklearnutilsextmath fastlogdet A
Compute logdetA for A symmetric
Equivalent to nplogndetA but more robust It returns Inf if detA is non positive or is not defined
Parameters
Arraylike The matrix

63918sklearnutilsextmath density
sklearnutilsextmath densitywkwargs
Compute density of a sparse vector
Parameters
warraylike The sparse vector
Returns
float The density of w between 0 and 1

Examples using sklearnutilsextmathdensity

- Classification of text documents using sparse features

63919sklearnutilsextmath weightedmode
sklearnutilsextmath weightedmode aaxis0
Returns an array of the weighted modal most common value in a
If there is more than one such value only the first is returned The bincount for the modal bins is also returned
This is an extension of the algorithm in scipystatsmode
Parameters
aarraylike ndimensional array of which to find modes
warraylike ndimensional array of weights for each value
axis int optional Axis along which to operate Default is 0 ie the first axis
Returns
vals ndarray Array of modal values
score ndarray Array of weighted counts for each mode
See also
scipystatsmode
Examples

639sklearnutils Utilities 2369

scikitlearn user guide Release 0213

```
from sklearnutilsextmath import weightedmode
```

```
x 4 1 4 2 4 2
```

```
weights 1 1 1 1 1 1
```

```
weightedmodex weights
```

```
array4 array3
```

The value 4 appears three times with uniform weights the result is simply the mode of the distribution

```
weights 1 3 05 15 1 2 deweight the 4s
```

```
weightedmodex weights
```

```
array2 array35
```

The value 2 has the highest score it appears twice with weights of 15 and 2 the sum of these is 35

63920sklearnutils genevenslices

sklearnutils genevenslices npacks nsamplesNone

Generator to create npacks slices going up to n

Parameters

nint

npacks int Number of slices to generate

nsamples int or None default None Number of samples Pass nsamples when the slices

are to be used for sparse matrix indexing slicing offtheend raises an exception while it

works for NumPy arrays

Yields

slice

Examples

```
from sklearnutils import genevenslices
```

```
listgenevenslices10 1
```

```
slice0 10 None
```

```
listgenevenslices10 10
```

```
slice0 1 None slice1 2 None slice9 10 None
```

```
listgenevenslices10 5
```

```
slice0 2 None slice2 4 None slice8 10 None
```

```
listgenevenslices10 3
```

```
slice0 4 None slice4 7 None slice7 10 None
```

63921sklearnutilsgraph singlesourceshortestpathlength

sklearnutilsgraph singlesourceshortestpathlength graph source cut

offNone

Return the shortest path length from source to all reachable nodes

Returns a dictionary of shortest path lengths keyed by target

Parameters

graph sparse matrix or 2D array preferably LIL matrix Adjacency matrix of the graph

2370 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

source integer Starting node for path

cutoff integer optional Depth to stop the search only paths of length cutoff are returned

Examples

```
from sklearnutilsgraph import singlesourceshortestpathlength
```

```
import numpy as np
```

```
graph nparray 0 1 0 0
```

```
1 0 1 0
```

```
0 1 0 1
```

```
0 0 1 0
```

```
listsortedinglesourceshortestpathlengthgraph 0items
```

```
0 0 1 1 2 2 3 3
```

```
graph npones6 6
```

```
listsortedinglesourceshortestpathlengthgraph 2items
```

```
0 1 1 1 2 0 3 1 4 1 5 1
```

```
63922sklearnutilsgraphshortestpath graphshortestpath
```

```
sklearnutilsgraphshortestpath graphshortestpath
```

Perform a shortestpath graph search on a positive directed or undirected graph

Parameters

distmatrix arraylike or sparse matrix shape NN Array of positive distances If vertex

i is connected to vertex j then distmatrixij gives the distance between the vertices If

vertex i is not connected to vertex j then distmatrixij 0

directed boolean if True then find the shortest path on a directed graph only progress from

a point to its neighbors not the other way around if False then find the shortest path on an

undirected graph the algorithm can progress from a point to its neighbors and vice versa

method string 'auto''FW''D' method to use Options are 'auto' attempt to choose the best

method for the current problem 'FW' FloydWarshall algorithm ON3 'D' Dijkstra's

algorithm with Fibonacci stacks OKlogNN2

Returns

Gnpndarray float shape NN Gij gives the shortest distance from point i to point j

along the graph

Notes

As currently implemented Dijkstra's algorithm does not work for graphs with directiondependent distances

when directed False ie if distmatrixij and distmatrixji are not equal and both are nonzero

method'D' will not necessarily yield the correct result

Also these routines have not been tested for graphs with negative distances Negative distances can lead to

infinite cycles that must be handled by specialized algorithms

63923sklearnutils indexable

sklearnutils indexable iterables

Make arrays indexable for crossvalidation

639sklearnutils Utilities 2371

scikitlearn user guide Release 0213

Checks consistent length passes through None and ensures that everything can be indexed by converting sparse matrices to csr and converting noniterable objects to arrays

Parameters

iterables lists dataframes arrays sparse matrices List of objects to ensure sliceability

63924sklearnutilsmetaestimators ifdelegatehasmethod

sklearnutilsmetaestimators ifdelegatehasmethod delegate

Create a decorator for methods that are delegated to a subestimator

This enables ducktyping by hasattr returning True according to the subestimator

Parameters

delegate string list of strings or tuple of strings Name of the subestimator that can be accessed as an attribute of the base object If a list or a tuple of names are provided the first subestimator that is an attribute of the base object will be used

Examples using sklearnutilsmetaestimatorsifdelegatehasmethod

- Inductive Clustering

63925sklearnutilsmulticlass typeoftarget y

sklearnutilsmulticlass typeoftarget y

Determine the type of data indicated by the target

Note that this type is the most specific type that can be inferred For example

- binary is more specific but compatible with multiclass
- multiclass of integers is more specific but compatible with continuous
- multilabelindicator is more specific but compatible with multiclassmultioutput

Parameters

yarraylike

Returns

targettype string One of

- ‘continuous’ yis an arraylike of floats that are not all integers and is 1d or a column vector
- ‘continuousmultioutput’ yis a 2d array of floats that are not all integers and both dimensions are of size 1
- ‘binary’ ycontains 2 discrete values and is 1d or a column vector
- ‘multiclass’ ycontains more than two discrete values is not a sequence of sequences and is 1d or a column vector
- ‘multiclassmultioutput’ yis a 2d array that contains more than two discrete values is not a sequence of sequences and both dimensions are of size 1
- ‘multilabelindicator’ yis a label indicator matrix an array of two dimensions with at least two columns and at most 2 unique values

2372 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

- ‘unknown’ yis arraylike but none of the above such as a 3d array sequence of sequences or an array of nonsequence objects

Examples

```
import numpy as np
typeoftarget01 06
continuous
typeoftarget1 1 1 1
binary
typeoftargeta b a
binary
typeoftarget10 20
binary
typeoftarget1 0 2
multiclass
typeoftarget10 00 30
multiclass
typeoftargeta b c
multiclass
typeoftargetnparray1 2 3 1
multiclassmultioutput
typeoftarget1 2
multiclassmultioutput
typeoftargetnparray15 20 30 16
continuousmultioutput
typeoftargetnparray0 1 1 1
multilabelindicator
63926sklearnutilsmulticlass ismultilabel
sklearnutilsmulticlass ismultilabel y
```

Check ifyis in a multilabel format

Parameters

ynumpy array of shape nsamples Target values

Returns

out bool Return True ifyis in a multilabel format else False

Examples

```
import numpy as np
from sklearnutilsmulticlass import ismultilabel
ismultilabel0 1 0 1
False
ismultilabel1 0 2
False
ismultilabelnparray1 0 0 0
True
ismultilabelnparray1 0 0
False
639sklearnutils Utilities 2373
```

scikitlearn user guide Release 0213

ismultilabelInparray1 0 0

True

63927sklearnutilsmulticlass uniquelabels

sklearnutilsmulticlass uniquelabels ys

Extract an ordered array of unique labels

We don't allow

- mix of multilabel and multiclass single label targets
- mix of label indicator matrix and anything else because there are no explicit labels
- mix of label indicator matrices of different sizes
- mix of string and integer labels

At the moment we also don't allow "multiclassmultioutput" input type

Parameters

ys arraylikes

Returns

out numpy array of shape nuniquelabels An ordered array of unique labels

Examples

from sklearnutilsmulticlass import uniquelabels

uniquelabels3 5 5 5 7 7

array3 5 7

uniquelabels1 2 3 4 2 2 3 4

array1 2 3 4

uniquelabels1 2 10 5 11

array 1 2 5 10 11

Examples using sklearnutilsmulticlassuniquelabels

•Confusion matrix

63928sklearnutils murmurhash332

sklearnutils murmurhash332

Compute the 32bit murmurhash3 of key at seed

The underlying implementation is MurmurHash3x8632 generating low latency 32bits hash suitable for im

plementing lookup tables Bloom filters count min sketch or feature hashing

Parameters

key int32 bytes unicode or ndarray with dtype int32 the physical object to hash

seed int optional default is 0 integer seed for the hashing algorithm

positive boolean optional default is False

2374 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

True the results is casted to an unsigned int from 0 to 2³² - 1

False the results is casted to a signed int from 2³¹ to 2³¹ - 1

63929sklearnutils resample

sklearnutils resample arrays options

Resample arrays or sparse matrices in a consistent way

The default strategy implements one step of the bootstrapping procedure

Parameters

arrays sequence of indexable datastructures Indexable datastructures can be arrays lists

dataframes or scipy sparse matrices with consistent first dimension

Returns

resampledarrays sequence of indexable datastructures Sequence of resampled copies of

the collections The original arrays are not impacted

Other Parameters

replace boolean True by default Implements resampling with replacement If False this will implement sliced random permutations

nsamples int None by default Number of samples to generate If left to None this is automatically set to the first dimension of the arrays If replace is False it should not be larger than the length of arrays

randomstate int RandomState instance or None optional defaultNone The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

stratify arraylike or None defaultNone If not None data is split in a stratified fashion using this as the class labels

See also

sklearnutilsshuffle

Examples

It is possible to mix sparse and dense arrays in the same run

```
X = nparray1 0 2 1 0 0
```

```
y = nparray0 1 2
```

```
from scipy.sparse import coomatrix
```

```
Xsparse = coomatrixX
```

```
from sklearnutils import resample
```

```
X Xsparse y = resampleX Xsparse y randomstate0
```

```
X
```

```
array1 0
```

```
2 1
```

```
1 0
```

6393sklearnutils Utilities 2375

scikitlearn user guide Release 0213

Xsparse

3x2 sparse matrix of type numpyfloat64

with 4 stored elements in Compressed Sparse Row format

Xsparsetoarray

array1 0

2 1

1 0

y

array0 1 0

resampley nsamples2 randomstate0

array0 1

Example using stratification

y 0 0 1 1 1 1 1 1 1

resampley nsamples5 replace False stratify

randomstate0

1 1 1 0 1

63930sklearnutils safeindexing

sklearnutils safeindexing Xindices

Return items or rows from X using indices

Allows simple indexing of lists or arrays

Parameters

Xarraylike sparsematrix list pandasDataFrame pandasSeries Data from which to sam

ple rows or items

indices arraylike of int Indices according to which X will be subsampled

Returns

subset Subset of X on first axis

Notes

CSR (Compressed Sparse Row) CSC and LIL sparse matrices are supported COO sparse matrices are not supported

63931sklearnutils safemask

sklearnutils safemask Xmask

Return a mask which is safe to use on X

Parameters

Xarraylike sparse matrix Data on which to apply mask

mask array Mask to be used on X

Returns

2376 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

mask

63932sklearnutils safesqr

sklearnutils safesqr XcopyTrue

Element wise squaring of arraylikes and sparse matrices

Parameters

Xarray like matrix sparse matrix

copy boolean optional default True Whether to create a copy of X and operate on it or to

perform inplace computation default behaviour

Returns

X 2 element wise square

63933sklearnutils shuffle

sklearnutils shufflearrays options

Shuffle arrays or sparse matrices in a consistent way

This is a convenience alias to resample arrays replaceFalse to do random permutations of the collections

Parameters

arrays sequence of indexable datastructures Indexable datastructures can be arrays lists

dataframes or scipy sparse matrices with consistent first dimension

Returns

shuffledarrays sequence of indexable datastructures Sequence of shuffled copies of the col

lections The original arrays are not impacted

Other Parameters

randomstate int RandomState instance or None optional defaultNone The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom

nsamples int None by default Number of samples to generate If left to None this is automatically set to the first dimension of the arrays

See also

sklearnutilsresample

Examples

It is possible to mix sparse and dense arrays in the same run

639sklearnutils Utilities 2377

scikitlearn user guide Release 0213

```
X nparray1 0 2 1 0 0
y nparray0 1 2
from scipysparse import coomatrix
Xsparse coomatrixX
from sklearnutils import shuffle
X Xsparse y shuffleX Xsparse y randomstate0
X
array0 0
2 1
1 0
Xsparse
3x2 sparse matrix of type numpyfloat64
with 3 stored elements in Compressed Sparse Row format
Xsparsetoarray
array0 0
2 1
1 0
y
array2 1 0
shuffle nsamples2 randomstate0
array0 1
```

Examples using sklearnutilsshuffle

- Model Complexity Influence
- Prediction Latency
- Color Quantization using KMeans
- Empirical evaluation of the impact of kmeans initialization
- Gradient Boosting regression
- Early stopping of Stochastic Gradient Descent

63934sklearnutilssparsefuncs incrmearvarianceaxis

sklearnutilssparsefuncs incrmearvarianceaxis Xaxislastmean lastvar lastn

Compute incremental mean and variance along an axix on a CSR (Compressed Sparse Row) or CSC matrix
lastmean lastvar are the statistics computed at the last step by this function Both must be initialized to 0
arrays of the proper size ie the number of features in X lastn is the number of samples encountered until
now

Parameters

XCSR or CSC sparse matrix shape nsamples nfeatures Input data

axis int either 0 or 1 Axis along which the axis should be computed

lastmean float array with shape nfeatures Array of featurewise means to update with the
new data X

2378 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

lastvar float array with shape nfeatures Array of featurewise var to update with the new data X

lastn int with shape nfeatures Number of samples seen so far excluded X

Returns

means float array with shape nfeatures Updated featurewise means

variances float array with shape nfeatures Updated featurewise variances

nint with shape nfeatures Updated number of seen samples

Notes

NaNs are ignored in the algorithm

63935sklearnutilssparsefuncs inplacecolumnscale

sklearnutilssparsefuncs inplacecolumnscale Xscale

Inplace column scaling of a CSCCSR matrix

Scale each feature of the data matrix by multiplying with specific scale provided by the caller assuming a

nsamples nfeatures shape

Parameters

XCSC or CSR (Compressed Sparse Row) matrix with shape nsamples nfeatures Matrix to normalize using the variance of the features

scale float array with shape nfeatures Array of precomputed featurewise values to use for

scaling

63936sklearnutilssparsefuncs inplacerowscale

sklearnutilssparsefuncs inplacerowscale Xscale

Inplace row scaling of a CSR (Compressed Sparse Row) or CSC matrix

Scale each row of the data matrix by multiplying with specific scale provided by the caller assuming a

nsamples nfeatures shape

Parameters

XCSR or CSC sparse matrix shape nsamples nfeatures Matrix to be scaled

scale float array with shape nfeatures Array of precomputed samplewise values to use for

scaling

63937sklearnutilssparsefuncs inplaceswaprow

sklearnutilssparsefuncs inplaceswaprow Xmn

Swaps two rows of a CSCCSR matrix inplace

Parameters

XCSR or CSC sparse matrix shapensamples nfeatures Matrix whose two rows are to

be swapped

639sklearnutils Utilities 2379

scikitlearn user guide Release 0213

mint Index of the row of X to be swapped

nint Index of the row of X to be swapped

63938sklearnutilssparsfuncs inplacecolumn

sklearnutilssparsfuncs inplacecolumn Xmn

Swaps two columns of a CSCCSR matrix inplace

Parameters

XCSR or CSC sparse matrix shapensamples nfeatures Matrix whose two columns are

to be swapped

mint Index of the column of X to be swapped

nint Index of the column of X to be swapped

63939sklearnutilssparsfuncs meanvarianceaxis

sklearnutilssparsfuncs meanvarianceaxis Xaxis

Compute mean and variance along an axis on a CSR (Compressed Sparse Row) or CSC matrix

Parameters

XCSR or CSC sparse matrix shape nsamples nfeatures Input data

axis int either 0 or 1 Axis along which the axis should be computed

Returns

means float array with shape nfeatures Featurewise means

variances float array with shape nfeatures Featurewise variances

63940sklearnutilssparsfuncs inplacecolumnscale

sklearnutilssparsfuncs inplacecolumnscale Xscale

Inplace column scaling of a CSR (Compressed Sparse Row) matrix

Scale each feature of the data matrix by multiplying with specific scale provided by the caller assuming a

nsamples nfeatures shape

Parameters

XCSR matrix with shape nsamples nfeatures Matrix to normalize using the variance of

the features

scale float array with shape nfeatures Array of precomputed featurewise values to use for

scaling

63941sklearnutilssparsfuncsfast inplacecolumnnormalizel1

sklearnutilssparsfuncsfast inplacecolumnnormalizel1

Inplace row normalize using the l1 norm

2380 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

63942sklearnutilssparsefuncsfast inplacecsrrownormalizel2

sklearnutilssparsefuncsfast inplacecsrrownormalizel2

Inplace row normalize using the l2 norm

63943sklearnutilsrandom samplewithoutreplacement

sklearnutilsrandom samplewithoutreplacement

Sample integers without replacement

Select nsamples integers from the set 0 npopulation without replacement

Parameters

npopulation int The size of the set to sample from

nsamples int The number of integer to sample

randomstate int RandomState instance or None optional defaultNone If int ran

domstate is the seed used by the random number generator If RandomState instance

randomstate is the random number generator If None the random number generator is

the RandomState instance used by nprandom

method "auto" "trackingselection" "reservoirsampling" or "pool" If method "auto"

the ratio of nsamples npopulation is used to determine which algorithm to use If ra

tio is between 0 and 001 tracking selection is used If ratio is between 001 and 099

numpyrandompermutation is used If ratio is greater than 099 reservoir sampling is used

The order of the selected integers is undefined If a random order is desired the selected

subset should be shuffled

If method "trackingselection" a set based implementation is used which is suitable for

nsamples npopulation

If method "reservoirsampling" a reservoir sampling algorithm is used which is suitable

for high memory constraint or when O nsamples Onpopulation The order of the

selected integers is undefined If a random order is desired the selected subset should be

shuffled

If method "pool" a pool based algorithm is particularly fast even faster than the tracking

selection method However a vector containing the entire population has to be initialized

If nsamples npopulation the reservoir sampling method is faster

Returns

out array of size nsamples The sampled subsets of integer The subset of selected integer

might not be randomized see the method argument

63944sklearnutilvalidation checkisfitted

sklearnutilvalidation checkisfitted estimator attributes msgNone

alloranybuiltin function all

Perform isfitted validation for estimator

Checks if the estimator is fitted by verifying the presence of "allorany" of the passed attributes and raises a

NotFittedError with the given message

Parameters

estimator estimator instance estimator instance for which the check is performed

639sklearnutils Utilities 2381

scikitlearn user guide Release 0213

attributes attribute names given as string or a listtuple of strings

Egcoef estimator coef

msg string The default error message is “This names instance is not fitted yet Call ‘fit’ with appropriate arguments before using this method”

For custom messages if “names” is present in the message string it is substituted for the estimator name

Eg “Estimator names must be fitted before sparsifying”

allorany callable all any default all Specify whether all or any of the given attributes must exist

Returns

None

Raises

NotFittedError If the attributes are not found

63945sklearnutilsvalidation checkmemory

sklearnutilsvalidation checkmemory memory

Check that memory is joblibMemorylike

joblibMemorylike means that memory can be converted into a joblibMemory instance typically a str denoting thelocation or has the same interface has a cache method

Parameters

memory None str or object with the joblibMemory interface

Returns

memory object with the joblibMemory interface

Raises

ValueError Ifmemory is not joblibMemorylike

63946sklearnutilsvalidation checksymmetric

sklearnutilsvalidation checksymmetric array tol1e10 raisewarningTrue

raiseexceptionFalse

Make sure that array is 2D square and symmetric

If the array is not symmetric then a symmetrized version is returned Optionally a warning or exception is raised if the matrix is not symmetric

Parameters

array ndarray or sparse matrix Input object to check convert Must be twodimensional and square otherwise a ValueError will be raised

tolfloat Absolute tolerance for equivalence of arrays Default 1E10

raisewarning boolean defaultTrue If True then raise a warning if conversion is required

raiseexception boolean defaultFalse If True then raise an exception if array is not sym
metric

2382 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Returns
arraysym ndarray or sparse matrix Symmetrized version of the input array ie the average of array and arraytranspose If sparse then duplicate entries are first summed and zeros are eliminated

63947sklearnutilvalidation columnnor1d
sklearnutilvalidation columnnor1d ywarnFalse
Ravel column or 1d numpy array else raises an error

Parameters
yarraylike
warn boolean default False To control display of warnings

Returns
yarray
63948sklearnutilvalidation hasfitparameter
sklearnutilvalidation hasfitparameter estimator parameter
Checks whether the estimator’s fit method supports the given parameter

Parameters
estimator object An estimator to inspect
parameter str The searched parameter
Returns
isparameter bool Whether the parameter was found to be a named parameter of the estimator’s fit method

Examples
from sklearnsvm import SVC (Support Vector Classification)
hasfitparameterSVC sampleweight
True

63949sklearnutiltesting assertin
sklearnutiltesting assertin member container msgNone
Just like selfassertTruea in b but with a nicer default message
63950sklearnutiltesting assertnotin
sklearnutiltesting assertnotin member container msgNone
Just like selfassertTruea not in b but with a nicer default message
639sklearnutils Utilities 2383

scikitlearn user guide Release 0213

63951sklearnutilstesting assertraisemessage

sklearnutilstesting assertraisemessage exceptions message function args
kwargs

Helper function to test the message raised in an exception

Given an exception a callable to raise the exception and a message string tests that the correct exception is raised and that the message is a substring of the error thrown Used to test that the specific message thrown during an exception is correct

Parameters

exceptions exception or tuple of exception An Exception object

message str The error message or a substring of the error message

function callable Callable object to raise error

args the positional arguments to function

kwargs the keyword arguments to function

63952sklearnutilstesting allestimators

sklearnutilstesting allestimators includemetaestimatorsNone includeootherNone

typefilterNone includedonttestNone

Get a list of all estimators from sklearn

This function crawls the module and gets all classes that inherit from BaseEstimator Classes that are defined in

testmodules are not included By default metaestimators such as GridSearchCV are also not included

Parameters

includemetaestimators boolean defaultFalse Deprecated ignored deprecated 021

includemetaestimators has been deprecated and has no effect in 021 and
will be removed in 023

includeoother boolean defaultFalse Deprecated ignored deprecated 021

includeoother has been deprecated and has not effect in 021 and will be removed
in 023

typefilter string list of string or None defaultNone Which kind of estimators should be

returned If None no filter is applied and all estimators are returned Possible values are

‘classifier’ ‘regressor’ ‘cluster’ and ‘transformer’ to get estimators only of these specific
types or a list of these to get the estimators that fit at least one of the types

includedonttest boolean defaultFalse Deprecated ignored deprecated 021

includedonttest has been deprecated and has no effect in 021 and will be
removed in 023

Returns

estimators list of tuples List of name class where name is the class name as string and

class is the actual type of the class

Utilities from joblib

2384 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

utilsparallelbackend backend njobs Change the default backend used by Parallel inside a with block

utilsregisterparallelbackend name fac

Register a new Parallel backend factory

sklearnutils parallelbackend

sklearnutils parallelbackend backend njobs1 backendparams

Change the default backend used by Parallel inside a with block

If backend is a string it must match a previously registered implementation using the registerparallelbackend function

By default the following backends are available

- ‘loky’ singlehost processbased parallelism used by default
- ‘threading’ singlehost threadbased parallelism
- ‘multiprocessing’ legacy singlehost processbased parallelism

‘loky’ is recommended to run functions that manipulate Python objects ‘threading’ is a lowoverhead alternative that is most efficient for functions that release the Global Interpreter Lock eg IObound code or CPUbound code in a few calls to native code that explicitly releases the GIL

In addition if the dask anddistributed Python packages are installed it is possible to use the ‘dask’ backend for better scheduling of nested parallel calls without oversubscription and potentially distribute parallel calls over a networked cluster of several hosts

Alternatively the backend can be passed directly as an instance

By default all available workers will be used njobs1 unless the caller passes an explicit value for the njobs parameter

This is an alternative to passing a backendbackendname argument to the Parallel class constructor

It is particularly useful when calling into library code that uses joblib internally but does not expose the backend argument in its own API (Application Programming Interface)

```
from operator import neg
with parallelbackendthreading
printParalleldelayednegi 1 foriinrange5
```

1 2 3 4 5

Warning this function is experimental and subject to change in a future version of joblib

New in version 010

sklearnutils registerparallelbackend

sklearnutils registerparallelbackend name factory makedefaultFalse

Register a new Parallel backend factory

The new backend can then be selected by passing its name as the backend argument to the Parallel class

Moreover the default backend can be overwritten globally by setting makedefaultTrue

The factory can be any callable that takes no argument and return an instance of ParallelBackendBase

Warning this function is experimental and subject to change in a future version of joblib

sklearnutils Utilities 2385

scikitlearn user guide Release 0213

New in version 010

640 Recently deprecated

6401 To be removed in 023

utilsMemory args kwargs

Attributes

utilsParallel args kwargs

Methods

sklearnutils Memory

Warning DEPRECATED

classsklearnutils Memoryargs kwargs

Attributes

cachedir

Methods

cache self func ignore verbose mmapmode Decorates the given function func to only compute its return value for input arguments not cached on disk

clear self warn Erase the complete cache directory

eval self func args kwargs Eval function func with arguments args and kwargs in the context of the memory

format self obj indent Return the formatted representation of the object

reducesize self Remove cache elements to make cache size fit in byteslimit

debug

warn

init args kwargs

DEPRECATED deprecated in version 0201 to be removed in version 023 Please import this function ality directly from joblib which can be installed with pip install joblib

cacheselffuncNone ignoreNone verboseNone mmapmodeFalse

Decorates the given function func to only compute its return value for input arguments not cached on disk

Parameters

2386 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

func callable optional The function to be decorated

ignore list of strings A list of arguments name to ignore in the hashing

verbose integer optional The verbosity mode of the function By default that of the memory object is used

mmapmode None 'r' 'r+' 'w' 'c' optional The memmapping mode used when loading from cache numpy arrays See numpyload for the meaning of the arguments

By default that of the memory object is used

Returns

decoratedfunc MemorizedFunc object The returned object is a MemorizedFunc object that is callable behaves like a function but offers extra methods for cache lookup and management See the documentation for joblibmemoryMemorizedFunc

clearselfwarnTrue

Erase the complete cache directory

evalselffuncargs kwargs

Eval function func with arguments args andkwargs in the context of the memory

This method works similarly to the builtin apply except that the function is called only if the cache is not up to date

formatselfobjindent0

Return the formatted representation of the object

reducesize self

Remove cache elements to make cache size fit in byteslimit

sklearnutils Parallel

Warning DEPRECATED

classsklearnutils Parallel args kwargs

Methods

call self iterable

dispatchnext self Dispatch more data for parallel processing

dispatchonebatch self iterator Prefetch the tasks for the next batch and dispatch them

format self obj indent Return the formatted representation of the object

printprogress self Display the process of the parallel execution only a fraction of time controlled by selfverbose

debug

retrieve

warn

init args kwargs

DEPRECATED deprecated in version 0201 to be removed in version 023 Please import this function

640 Recently deprecated 2387

scikitlearn user guide Release 0.21.3

ality directly from joblib which can be installed with pip install joblib

dispatchnext self

Dispatch more data for parallel processing

This method is meant to be called concurrently by the multiprocessing callback We rely on the thread safety of dispatchonebatch to protect against concurrent consumption of the unprotected iterator

dispatchonebatch selfiterator

Prefetch the tasks for the next batch and dispatch them

The effective size of the batch is computed here If there are no more jobs to dispatch return False else return True

The iterator consumption and dispatching is protected by the same lock so calling this function should be thread safe

formatselfobjindent0

Return the formatted representation of the object

printprogress self

Display the progress of the parallel execution only a fraction of time controlled by selfverbose

utilscpucount DEPRECATED deprecated in version 0.20.1 to be removed in version 0.23

utilsdelayed function checkpickle DEPRECATED deprecated in version 0.20.1 to be removed in version 0.23

metricscalinskiharabaszscore X labels DEPRECATED Function 'calinskiharabaszscore' has been renamed to 'calinskiharabaszscore' and will be removed in version 0.23

metricsjaccardsimilarityscore ytrue

ypredJaccard similarity coefficient score

linearmodellogisticregressionpath X

yDEPRECATED logisticregressionpath was deprecated in version 0.21 and will be removed in version 0.23

sklearnutils cpucount

Warning DEPRECATED

sklearnutils cpucount

DEPRECATED deprecated in version 0.20.1 to be removed in version 0.23 Please import this functionality directly from joblib which can be installed with pip install joblib

Return the number of CPUs

sklearnutils delayed

Warning DEPRECATED

sklearnutils delayedfunction checkpickleNone

DEPRECATED deprecated in version 0.20.1 to be removed in version 0.23 Please import this functionality directly from joblib which can be installed with pip install joblib

2388 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Decorator used to capture the arguments of a function

sklearnmetrics calinskiharabazscore

Warning DEPRECATED

sklearnmetrics calinskiharabazscore Xlabels

DEPRECATED Function 'calinskiharabazscore' has been renamed to 'calinskiharabaszscore' and will be removed in version 023

sklearnmetrics jaccardsimilarityscore

Warning DEPRECATED

sklearnmetrics jaccardsimilarityscore ytrue ypred normalizeTrue sampleweightNone

Jaccard similarity coefficient score

Deprecated since version 021 This is deprecated to be removed in 023 since its handling of binary and multiclass inputs was broken jaccardscore has an API (Application Programming Interface) that is consistent with precisionscore fscore etc

Read more in the User Guide

Parameters

ytrue 1d arraylike or label indicator array sparse matrix Ground truth correct labels

ypred 1d arraylike or label indicator array sparse matrix Predicted labels as returned by a classifier

normalize bool optional defaultTrue If False return the sum of the Jaccard similarity coefficient over the sample set Otherwise return the average of Jaccard similarity coefficient

sampleweight arraylike of shape nsamples optional Sample weights

Returns

score float Ifnormalize True return the average Jaccard similarity coefficient else it returns the sum of the Jaccard similarity coefficient over the sample set

The best performance is 1 with normalize True and the number of samples with normalize False

See also

accuracy score hammingloss zeroone loss

Notes

In binary and multiclass classification this function is equivalent to the accuracy score It differs in the multilabel classification problem

640 Recently deprecated 2389

References

1

sklearnlinear model logisticregressionpath

Warning DEPRECATED

sklearnlinear model logisticregressionpath Xy posclassNone Cs10

fitinterceptTrue maxiter100

tol000001 verbose0 solver'lbfgs'

coefNone classweightNone

dualFalse penalty'l2' inter

ceptscaling10 multiclass'warn'

randomstateNone checkinputTrue

maxsquaredsumNone sam

pleweightNone l1ratioNone

DEPRECATED logisticregressionpath was deprecated in version 021 and will be removed in version 0230

Compute a Logistic Regression model for a list of regularization parameters

This is an implementation that uses the result of the previous model to speed up computations along the

set of solutions making it faster than sequentially calling LogisticRegression for the different parameters

Note that there will be no speedup with liblinear solver since it does not handle warmstarting

Deprecated since version 021 logisticregressionpath was deprecated in version 021 and

will be removed in 023

Read more in the User Guide

Parameters

Xarraylike or sparse matrix shape nsamples nfeatures

Input data

yarraylike shape nsamples or nsamples ntargets Input data target values

posclass int None The class with respect to which we perform a onevsall fit If None

then it is assumed that the given problem is binary

Csint arraylike shape ncs List of values for the regularization parameter or integer

specifying the number of regularization parameters that should be used In this case the

parameters will be chosen in a logarithmic scale between 1e4 and 1e4

fitintercept bool Whether to fit an intercept for the model In this case the shape of the

returned array is ncs nfeatures 1

maxiter int Maximum number of iterations for the solver

tolfloat Stopping criterion For the newtoncg and lbfgs solvers the iteration will stop

whenmaxgi i 1 n tol wheregi is the ith component

of the gradient

verbose int For the liblinear and lbfgs solvers set verbose to any positive number for

verbosity

solver 'lbfgs' 'newtoncg' 'liblinear' 'sag' 'saga' Numerical solver to use

scikitlearn user guide Release 0213

coef arraylike shape nfeatures default None Initialization value for coefficients of logistic regression Useless for liblinear solver

classweight dict or 'balanced' optional Weights associated with classes in the form classlabel weight If not given all classes are supposed to have weight one

The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as nsamples nclasses npbincount

Note that these weights will be multiplied with sampleweight passed through the fit method if sampleweight is specified

dual bool Dual or primal formulation Dual formulation is only implemented for l2 penalty with liblinear solver Prefer dualFalse when nsamples nfeatures

penalty str 'l1' 'l2' or 'elasticnet' Used to specify the norm used in the penalization The 'newtoncg' 'sag' and 'lbfgs' solvers support only l2 penalties 'elasticnet' is only supported by the 'saga' solver

interceptscaling float default 1 Useful only when the solver 'liblinear' is used and selffitintercept is set to True In this case x becomes x selfinterceptscaling

ie a "synthetic" feature with constant value equal to interceptscaling is appended to the instance vector The intercept becomes interceptscaling syntheticfeatureweight

Note the synthetic feature weight is subject to l1l2 regularization as all other features To lessen the effect of regularization on synthetic feature weight and therefore on the intercept interceptscaling has to be increased

multiclass str 'ovr' 'multinomial' 'auto' default 'ovr' If the option chosen is 'ovr' then a binary problem is fit for each label For 'multinomial' the loss minimised is the multinomial loss fit across the entire probability distribution even when the data is binary 'multinomial' is unavailable when solver'liblinear' 'auto' selects 'ovr' if the data is binary or if solver'liblinear' and otherwise selects 'multinomial'

New in version 018 Stochastic Average Gradient descent solver for 'multinomial' case Changed in version 020 Default will change from 'ovr' to 'auto' in 022

randomstate int RandomState instance or None optional default None The seed of the pseudo random number generator to use when shuffling the data If int randomstate is the seed used by the random number generator If RandomState instance randomstate is the random number generator If None the random number generator is the RandomState instance used by nprandom Used when solver 'sag' or 'liblinear'

checkinput bool default True If False the input arrays X and y will not be checked maxsquaredsum float default None Maximum squared sum of X over samples Used only in SAG solver If None it will be computed going through all the samples The value should be precomputed to speed up cross validation

sampleweight arraylike shapensamples optional Array of weights that are assigned to individual samples If not provided then each sample is given unit weight

l1ratio float or None optional defaultNone The ElasticNet mixing parameter with0 l1ratio 1 Only used if penaltyelasticnet Setting l1ratio0 is equivalent to using penaltyl2 while setting l1ratio1 is equivalent to using penaltyl1 For0 l1ratio 1 the penalty is a combination of L1 and L2

640 Recently deprecated 2391

scikitlearn user guide Release 0213

Returns

coefs ndarray shape ncs nfeatures or ncs nfeatures 1
List of coefficients for the Logistic Regression model If fitintercept is set to True
then the second dimension will be nfeatures 1 where the last item represents
the intercept For multiclassmultinomial the shape is nclasses ncs
nfeatures or nclasses ncs nfeatures 1
Csndarray Grid of Cs used for crossvalidation
niter array shape ncs Actual number of iteration for each Cs

Notes

You might get slightly different results with the solver liblinear than with the others since this uses LIBLINEAR
which penalizes the intercept
Changed in version 019 The “copy” parameter was removed
ensemblepartialdependence
partialdependence DEPRECATED The function ensem
blepartialdependence has been deprecated in favour
of inspectionpartialdependence in 021 and will be
removed in 023
ensemblepartialdependence
plotpartialdependence DEPRECATED The function ensem
bleplotpartialdependence has been deprecated in
favour of sklearninspectionplotpartialdependence in
021 and will be removed in 023
sklearnensemblepartialdependence partialdependence
sklearnensemblepartialdependence partialdependence gbrt targetvariables
gridNone XNone
percentiles005 095
gridresolution100
DEPRECATED The function ensemblepartialdependence has been deprecated in favour of inspec
tionpartialdependence in 021 and will be removed in 023
Partial dependence of targetvariables
Partial dependence plots show the dependence between the joint values of the
targetvariables and the function represented by the gbrt
Read more in the User Guide
Deprecated since version 021 This function was deprecated in version 021 in favor of sklearn
inspectionpartialdependence and will be removed in 023
Parameters
gbrt BaseGradientBoosting
A fitted gradient boosting model
targetvariables arraylike dtypeint The target features for which the partial depen
dency should be computed size should be smaller than 3 for visual renderings
2392 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

grid arraylike shapenpoints lentargetvariables The grid of targetvariables values for which the partial dependency should be evaluated either grid orXmust be specified

Xarraylike shapensamples nfeatures The data on which gbrt was trained It is used to generate a grid for thetargetvariables Thegrid comprises gridresolution equally spaced points between the two percentiles percentiles low high default005 095 The lower and upper percentile used create the extreme values for the grid Only ifXis not None

gridresolution int default100 The number of equally spaced points on the grid

Returns

pdp array shapenclasses npoints

The partial dependence function evaluated on the grid For regression and binary classification nclasses1

axes seq of ndarray or None The axes with which the grid has been created or None if the grid has been given

Examples

```
samples 0 0 2 1 0 0
labels 0 1
from sklearnensemble import GradientBoostingClassifier
gb GradientBoostingClassifierrandomstate0fitsamples labels
kwargs dictXsamples percentiles0 1 gridresolution2
partialdependencegb 0 kwargs
array452 452 array 0 1
sklearnensemblepartialdependence plotpartialdependence
sklearnensemblepartialdependence plotpartialdependence gbrtXfeatures featurenamesNone
labelNone
ncols3
gridresolution100
percentiles005
095 njobsNone
verbose0
axNone
linekwNone
contourkwNone
figkw
```

DEPRECATED The function ensembleplotpartialdependence has been deprecated in favour of sklearninspectionplotpartialdependence in 021 and will be removed in 023

Partial dependence plots for features

Thefeatures plots are arranged in a grid with ncols columns Twoway partial dependence plots are plotted as contour plots

Read more in the User Guide

640 Recently deprecated 2393

scikitlearn user guide Release 0213

Deprecated since version 021 This function was deprecated in version 021 in favor of sklearn inspectionplotpartialdependence and will be removed in 023

Parameters

gbrt BaseGradientBoosting

A fitted gradient boosting model

Xarraylike shapensamples nfeatures The data on which gbrt was trained

features seq of ints strings or tuples of ints or strings If seqi is an int or a tuple with one int value a oneway PDP is created if seqi is a tuple of two ints a two way PDP is created If featurenames is specified and seqi is an int seqi must be lenfeaturenames If seqi is a string featurenames must be specified and seqi must be in featurenames

featurenames seq of str Name of each feature featurenamesi holds the name of the feature with index i

label object The class label for which the PDPs should be computed Only if gbrt is a multiclass model Must be in gbrtclasses

ncols int The number of columns in the grid plot default 3

gridresolution int default100 The number of equally spaced points on the axes

percentiles low high default005 095 The lower and upper percentile used to create the extreme values for the PDP axes

njobs int or None optional defaultNone None means 1 unless in a joblib parallelbackend context1means using all processors See Glossary for more details

verbose int Verbose output during PD computations Defaults to 0

axMatplotlib axis object default None An axis object onto which the plots will be drawn

linekw dict Dict with keywords passed to the matplotlib.pyplot.plot call For oneway partial dependence plots

contourkw dict Dict with keywords passed to the matplotlib.pyplot.plot call For twoway partial dependence plots

figkw dict Dict with keywords passed to the figure call Note that all keywords not recognized above will be automatically included here

Returns

figfigure

The Matplotlib Figure object

axs seq of Axis objects A seq of Axis objects one for each subplot

Examples

```
from sklearn.datasets import makefriedman1
from sklearn.ensemble import GradientBoostingRegressor
X y = makefriedman1
clf = GradientBoostingRegressor(n_estimators=10)
clf.fit(X y)
```

2394 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213
fig axs plotpartialdependenceclf X 0 0 1

6402 To be removed in 022
covarianceGraphLasso args kwargs Sparse inverse covariance estimation with an l1penalized estimator
covarianceGraphLassoCV args kwargs Sparse inverse covariance w crossvalidated choice of the l1 penalty
preprocessingImputer args kwargs Imputation transformer for completing missing values
utilstestingmockmldataurlopen args
Object that mocks the urlopen function to fake requests to
mldata
sklearncovariance GraphLasso
Warning DEPRECATED
classsklearncovariance GraphLasso args kwargs
Sparse inverse covariance estimation with an l1penalized estimator
This class implements the Graphical Lasso algorithm
Read more in the User Guide
Parameters
alpha positive float default 0.01 The regularization parameter the higher alpha the more regularization the sparser the inverse covariance
mode 'cd' 'lars' default 'cd' The Lasso solver to use coordinate descent or LARS Use LARS for very sparse underlying graphs where p > n Elsewhere prefer cd which is more numerically stable
tol positive float default 1e4 The tolerance to declare convergence if the dual gap goes below this value iterations are stopped
enettol positive float optional The tolerance for the elastic net solver used to calculate the descent direction This parameter controls the accuracy of the search direction for a given column update not of the overall parameter estimate Only used for mode 'cd'
maxiter integer default 100 The maximum number of iterations
verbose boolean default False If verbose is True the objective function and dual gap are plotted at each iteration
assumecentered boolean default False If True data are not centered before computation Useful when working with data whose mean is almost but not exactly zero If False data are centered before computation
Attributes
covariance arraylike shape nfeatures nfeatures Estimated covariance matrix
precision arraylike shape nfeatures nfeatures Estimated pseudo inverse matrix
niter int Number of iterations run
640 Recently deprecated 2395

scikitlearn user guide Release 0213

See also

graphlasso GraphLassoCV

Methods

errornorm self compcov norm scaling Computes the Mean Squared Error between two covariance estimators

fitself X y Fits the GraphicalLasso model to X

getparams self deep Get parameters for this estimator

getprecision self Getter for the precision matrix

mahalanobis self X Computes the squared Mahalanobis distances of given observations

score self Xtest y Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

setparams self params Set the parameters of this estimator

init args kwargs

DEPRECATED The 'GraphLasso' was renamed to 'GraphicalLasso' in version 020 and will be removed in 022

errornorm selfcompcov norm'frobenius' scalingTrue squaredTrue

Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius norm

Parameters

compcov arraylike shape nfeatures nfeatures The covariance to compare with

norm str The type of norm used to compute the error Available error types 'frobenius'

default sqrttrAtA 'spectral' sqrtmaxeigenvaluesAtA where A is the error

compcov selfcovariance

scaling bool If True default the squared error norm is divided by nfeatures If False the squared error norm is not rescaled

squared bool Whether to compute the squared error norm or the error norm If True

default the squared error norm is returned If False the error norm is returned

Returns

The Mean Squared Error in the sense of the Frobenius norm between

self andcompcov covariance estimators

fitselfXyNone

Fits the GraphicalLasso model to X

Parameters

Xndarray shape nsamples nfeatures Data from which to compute the covariance estimate

yignored

getparams selfdeepTrue

Get parameters for this estimator

2396 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

Parameters

deep boolean optional If True will return the parameters for this estimator and contained subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances of the which we compute Observations are assumed to be drawn from the same distribution than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the likelihood where nsamples is the number of samples and nfeatures is the number of features Xtest is assumed to be drawn from the same distribution than the data used in fit including centering

y not used present for API (Application Programming Interface) consistence purpose

Returns

resfloat The likelihood of the data set with selfcovariance as an estimator of its covariance matrix

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

sklearn covariance GraphLassoCV

Warning DEPRECATED

640 Recently deprecated 2397

scikitlearn user guide Release 0213

classssklearnncovariance GraphLassoCV args kwargs

Sparse inverse covariance w crossvalidated choice of the l1 penalty

See glossary entry for crossvalidation estimator

This class implements the Graphical Lasso algorithm

Read more in the User Guide

Parameters

alphas integer or list positive float optional If an integer is given it fixes the number of points on the grids of alpha to be used If a list is given it gives the grid to be used See the notes in the class docstring for more details

nrefinements strictly positive integer The number of times the grid is refined Not used if explicit values of alphas are passed

cvint crossvalidation generator or an iterable optional Determines the crossvalidation splitting strategy Possible inputs for cv are

- None to use the default 3fold crossvalidation
- integer to specify the number of folds
- CV (Cross-Validation) splitter
- An iterable yielding train test splits as arrays of indices

For integerNone inputs KFold is used

Refer User Guide for the various crossvalidation strategies that can be used here

Changed in version 020 cvdefault value if None will change from 3fold to 5fold in v022

tolpositive float optional The tolerance to declare convergence if the dual gap goes below this value iterations are stopped

enettol positive float optional The tolerance for the elastic net solver used to calculate the descent direction This parameter controls the accuracy of the search direction for a given column update not of the overall parameter estimate Only used for mode'cd'

maxiter integer optional Maximum number of iterations

mode 'cd' 'lars' The Lasso solver to use coordinate descent or LARS Use LARS for very sparse underlying graphs where number of features is greater than number of samples Elsewhere prefer cd which is more numerically stable

njobs int or None optional defaultNone number of jobs to run in parallel None means 1 unless in a joblibparallelbackend context1means using all processors See Glossary for more details

verbose boolean optional If verbose is True the objective function and duality gap are printed at each iteration

assumecentered Boolean If True data are not centered before computation Useful when working with data whose mean is almost but not exactly zero If False data are centered before computation

Attributes

covariance numpyndarray shape nfeatures nfeatures Estimated covariance matrix

precision numpyndarray shape nfeatures nfeatures Estimated precision matrix inverse covariance

2398 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

alpha float Penalization parameter selected

cvalphas list of float All penalization parameters explored

gridscores 2D numpyndarray nalphas nfolds Loglikelihood score on leftout data

across folds

niter int Number of iterations run for the optimal alpha

See also

graphlasso GraphLasso

Notes

The search for the optimal penalization parameter alpha is done on an iteratively refined grid first the cross validated scores on a grid are computed then a new refined grid is centered around the maximum and so on One of the challenges which is faced here is that the solvers can fail to converge to a wellconditioned estimate The corresponding values of alpha then come out as missing values but the optimum may be close to these missing values

Methods

errornorm self compcov norm scaling Computes the Mean Squared Error between two covari
ance estimators

fitself X y Fits the GraphicalLasso covariance model to X

getparams self deep Get parameters for this estimator

getprecision self Getter for the precision matrix

mahalanobis self X Computes the squared Mahalanobis distances of given
observations

score self Xtest y Computes the loglikelihood of a Gaussian data set with
selfcovariance as an estimator of its covari

ance matrix

setparams self params Set the parameters of this estimator

init args kwargs

DEPRECATED The ‘GraphLassoCV’ was renamed to ‘GraphicalLassoCV’ in version 020 and will be
removed in 022

errornorm selfcompcov norm‘frobenius’ scalingTrue squaredTrue

Computes the Mean Squared Error between two covariance estimators In the sense of the Frobenius
norm

Parameters

compcov arraylike shape nfeatures nfeatures The covariance to compare with

norm str The type of norm used to compute the error Available error types ‘frobenius’

default sqrttrAtA ‘spectral’ sqrtmaxeigenvaluesAtA where A is the error

compcov selfcovariance

scaling bool If True default the squared error norm is divided by nfeatures If False
the squared error norm is not rescaled

squared bool Whether to compute the squared error norm or the error norm If True
640 Recently deprecated 2399

scikitlearn user guide Release 0213

default the squared error norm is returned If False the error norm is returned

Returns

The Mean Squared Error in the sense of the Frobenius norm between

self andcompcov covariance estimators

fitselfXyNone

Fits the GraphicalLasso covariance model to X

Parameters

Xndarray shape nsamples nfeatures Data from which to compute the covariance es

timate

yignored

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

getprecision self

Getter for the precision matrix

Returns

precision arraylike The precision matrix associated to the current covariance object

mahalanobis selfX

Computes the squared Mahalanobis distances of given observations

Parameters

Xarraylike shape nsamples nfeatures The observations the Mahalanobis distances

of the which we compute Observations are assumed to be drawn from the same distribu

tion than the data used in fit

Returns

dist array shape nsamples Squared Mahalanobis distances of the observations

scoreselfXtest yNone

Computes the loglikelihood of a Gaussian data set with selfcovariance as an estimator of its

covariance matrix

Parameters

Xtest arraylike shape nsamples nfeatures Test data of which we compute the

likelihood where nsamples is the number of samples and nfeatures is the number of

features Xtest is assumed to be drawn from the same distribution than the data used in

fit including centering

ynot used present for API (Application Programming Interface) consistence purpose

Returns

resfloat The likelihood of the data set with selfcovariance as an estimator of its

covariance matrix

2400 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

setparams selfparams

Set the parameters of this estimator

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it’s possible to update each component of a nested object

Returns

self

sklearnpreprocessing Imputer

Warning DEPRECATED

classsklearnpreprocessing Imputerargs kwargs

Imputation transformer for completing missing values

Read more in the User Guide

Parameters

missingvalues integer or “NaN” optional default“NaN” The placeholder for the missing values All occurrences of missingvalues will be imputed For missing values

encoded as npnan use the string value “NaN”

strategy string optional default“mean” The imputation strategy

- If “mean” then replace missing values using the mean along the axis
 - If “median” then replace missing values using the median along the axis
 - If “mostfrequent” then replace missing using the most frequent value along the axis
- axis integer optional default0 The axis along which to impute

• Ifaxis0 then impute along columns

• Ifaxis1 then impute along rows

verbose integer optional default0 Controls the verbosity of the imputer

copy boolean optional defaultTrue If True a copy of X will be created If False imputation will be done inplace whenever possible Note that in the following cases a new copy will always be made even if copyFalse

- If X is not an array of floating values
- If X is sparse and missingvalues0
- Ifaxis0 and X is encoded as a CSR (Compressed Sparse Row) matrix
- Ifaxis1 and X is encoded as a CSC matrix

Attributes

statistics array of shape nfeatures The imputation fill value for each feature if axis 0
640 Recently deprecated 2401

Notes

- Whenaxis0 columns which only contained missing values at fit are discarded upon transform
- Whenaxis1 an exception is raised if there are rows for which it is not possible to fill in the missing values eg because they only contain missing values

Methods

fitself X y Fit the imputer on X

fittransform self X y Fit to data then transform it

getparams self deep Get parameters for this estimator

setparams self params Set the parameters of this estimator

transform self X Impute all missing values in X

init args kwargs

DEPRECATED Imputer was deprecated in version 020 and will be removed in 022 Import im

puteSimpleImputer from sklearn instead

fitselfXyNone

Fit the imputer on X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures Input data where

nsamples is the number of samples and nfeatures is the number of features

Returns

self Imputer

fittransform selfXyNone fitparams

Fit to data then transform it

Fits transformer to X and y with optional parameters fitparams and returns a transformed version of X

Parameters

Xnumpy array of shape nsamples nfeatures Training set

ynumpy array of shape nsamples Target values

Returns

Xnew numpy array of shape nsamples nfeaturesnew Transformed array

getparams selfdeepTrue

Get parameters for this estimator

Parameters

deep boolean optional If True will return the parameters for this estimator and contained

subobjects that are estimators

Returns

params mapping of string to any Parameter names mapped to their values

setparams selfparams

Set the parameters of this estimator

scikitlearn user guide Release 0213

The method works on simple estimators as well as on nested objects such as pipelines The latter have parameters of the form componentparameter so that it's possible to update each component of a nested object

Returns

self

transform selfX

Impute all missing values in X

Parameters

Xarraylike sparse matrix shape nsamples nfeatures The input data to complete

sklearnutilstesting mockmldataurlopen

Warning DEPRECATED

classsklearnutilstesting mockmldataurlopen args kwargs

Object that mocks the urlopen function to fake requests to mldata

When requesting a dataset with a name that is in mockdatasets this object creates a fake dataset in a StringIO

object and returns it Otherwise it raises an HTTPError

Deprecated since version 020 Will be removed in version 022

Parameters

mockdatasets dict A dictionary of datasetname datadict or datasetname datadict

ordering datadict itself is a dictionary of columnname dataarray and

ordering is a list of columnnames to determine the ordering in the data set see

fakemldata for details

Methods

call self urlname

Parameters

init args kwargs

DEPRECATED deprecated in version 020 to be removed in version 022

covariancegraphlasso empcov alpha DEPRECATED The 'graphlasso' was renamed to

'graphicallasso' in version 020 and will be removed in

022

datasetsfetchmldata dataname DEPRECATED fetchmldata was deprecated in version

020 and will be removed in version 022

datasetsmldatafilename dataname DEPRECATED mldatafilename was deprecated in ver

sion 020 and will be removed in version 022

640 Recently deprecated 2403

scikitlearn user guide Release 0213

sklearn covariance graphlasso

Warning DEPRECATED

sklearn covariance graphlasso empcov alpha covinitNone mode'cd' tol00001

enettol00001 maxiter100 verboseFalse re

turncostsFalse eps2220446049250313e16 re

turnniterFalse

DEPRECATED The 'graphlasso' was renamed to 'graphicallasso' in version 020 and will be removed in

022

l1penalized covariance estimator

Read more in the User Guide

Parameters

empcov 2D ndarray shape nfeatures nfeatures

Empirical covariance from which to compute the covariance estimate

alpha positive float The regularization parameter the higher alpha the more regularization the sparser the inverse covariance

covinit 2D array nfeatures nfeatures optional The initial guess for the covariance

mode 'cd' 'lars' The Lasso solver to use coordinate descent or LARS Use LARS

for very sparse underlying graphs where $p \gg n$ Elsewhere prefer cd which is more numerically stable

tol positive float optional The tolerance to declare convergence if the dual gap goes below this value iterations are stopped

enettol positive float optional The tolerance for the elastic net solver used to calculate the descent direction This parameter controls the accuracy of the search direction for a given column update not of the overall parameter estimate Only used for mode'cd'

maxiter integer optional The maximum number of iterations

verbose boolean optional If verbose is True the objective function and dual gap are printed at each iteration

returncosts boolean optional If returncosts is True the objective function and dual gap at each iteration are returned

eps float optional The machineprecision regularization in the computation of the Cholesky diagonal factors Increase this for very illconditioned systems

turnniter bool optional Whether or not to return the number of iterations

Returns

covariance 2D ndarray shape nfeatures nfeatures

The estimated covariance matrix

precision 2D ndarray shape nfeatures nfeatures The estimated sparse precision matrix

2404 Chapter 6 API (Application Programming Interface) Reference

scikitlearn user guide Release 0213

costs list of objective dualgap pairs The list of values of the objective function and the

dual gap at each iteration Returned only if returncosts is True

niter int Number of iterations Returned only if returnniter is set to True

Notes

The algorithm employed to solve this problem is the GLasso algorithm from the Friedman 2008 Biostatistics paper It is the same algorithm as in the R glasso package

One possible difference with the glasso R package is that the diagonal coefficients are not penalized

sklearn.datasets.fetch_mldata

Warning DEPRECATED

sklearn.datasets.fetch_mldata dataname targetname'label' dataname'data' trans

posedata=True datahome=None

DEPRECATED fetch_mldata was deprecated in version 020 and will be removed in version 022 Please use

fetch_openml

Fetch an mldata.org data set

mldata.org is no longer operational

If the file does not exist yet it is downloaded from mldata.org

mldata.org does not have an enforced convention for storing data or naming the columns in a data

set The default behavior of this function works well with the most common cases

1 data values are stored in the column 'data' and target values in the column 'label'

2 alternatively the first column stores target values and the second data values

3 the data array is stored as nfeatures x nsamples and thus needs to be transposed to

match the sklearn standard

Keyword arguments allow to adapt these defaults to specific data sets see parameters

targetname dataname transposedata and the examples below

mldata.org data sets may have multiple columns which are stored in the Bunch object with their

original name

Deprecated since version 020 Will be removed in version 022

Parameters

dataname str

Name of the data set on mldata.org eg "leukemia" "Whistler Daily Snowfall" etc

The raw name is automatically converted to a mldata.org URL

targetname optional default 'label' Name or index of the column containing the target

values

dataname optional default 'data' Name or index of the column containing the data

transposedata optional default True If True transpose the downloaded data array

640 Recently deprecated 2405

scikitlearn user guide Release 0213

datahome optional default None Specify another download and cache folder for the data sets By default all scikitlearn data is stored in 'scikitlearndata' subfolders

Returns

data Bunch Dictionarylike object the interesting attributes are 'data' the data to learn 'target' the classification labels 'DESCR' the full description of the dataset and 'COLNAMES' the original names of the dataset columns

sklearn.datasets.mldatafilename

Warning DEPRECATED

sklearn.datasets.mldatafilename dataname

DEPRECATED mldatafilename was deprecated in version 020 and will be removed in version 022 Please use fetchopenml

Convert a raw name for a data set in a mldataorg filename

Deprecated since version 020 Will be removed in version 022

Parameters

dataname str Name of dataset

Returns

fname str The converted dataname

2406 Chapter 6 API (Application Programming Interface) Reference

CHAPTER
SEVEN
DEVELOPER’S GUIDE
71 Contributing

This project is a community effort and everyone is welcome to contribute
The project is hosted on <https://github.com/scikitlearn/scikitlearn>
The decision making process and governance structure of scikitlearn is laid out in the governance document [Scikit learn governance and decisionmaking](#)
Scikitlearn is somewhat selective when it comes to adding new algorithms and the best way to contribute and to help the project is to start working on known issues See [Issues for New Contributors](#) to get started
Our community our values
We are a community based on openness and friendly didactic discussions
We aspire to treat everybody equally and value their contributions
Decisions are made based on technical merit and consensus
Code is not the only way to help the project Reviewing pull requests answering questions to help others on mailing lists or issues organizing and teaching tutorials working on the website improving the documentation are all priceless contributions
We abide by the principles of openness respect and consideration of others of the Python Software Foundation <https://www.python.org/psf/codeofconduct>
In case you experience issues using this package do not hesitate to submit a ticket to the GitHub issue tracker You are also welcome to post feature requests or pull requests
711 Ways to contribute
There are many ways to contribute to scikitlearn with the most common ones being contribution of code or documentation to the project Improving the documentation is no less important than improving the library itself If you find a typo in the documentation or have made improvements do not hesitate to send an email to the mailing list or preferably submit a GitHub pull request Full documentation can be found under the doc directory
But there are many other ways to help In particular answering queries on the issue tracker investigating bugs and reviewing other developers’ pull requests are very valuable contributions that decrease the burden on the project maintainers
2407

scikitlearn user guide Release 0213

Another way to contribute is to report issues you’re facing and give a “thumbs up” on issues that others reported and that are relevant to you It also helps us if you spread the word reference the project from your blog and articles link to it from your website or simply star to say “I use it”

In case a contributionissue involves changes to the API (Application Programming Interface) principles or changes to dependencies or supported versions must be backed by a Enhancement proposals SLEPs where a SLEP must be submitted as a pullrequest to enhance the project. Enhancement proposals using the SLEP template and follows the decisionmaking process outlined in Scikitlearn governance and decisionmaking

Contributing to related projects

Scikitlearn thrives in an ecosystem of several related projects which also may have relevant issues to work on including smaller projects such as

- scikitlearncontrib
- joblib
- sphinxgallery
- numpydoc
- liacarff

and larger projects

- numpy
- scipy
- matplotlib
- and so on

Look for issues marked “help wanted” or similar Helping these projects may help Scikitlearn too See also Related Projects

712 Submitting a bug report or a feature request

We use GitHub issues to track all bugs and feature requests feel free to open an issue if you have found a bug or wish to see a feature implemented

In case you experience issues using this package do not hesitate to submit a ticket to the Bug Tracker You are also welcome to post feature requests or pull requests

It is recommended to check that your issue complies with the following rules before submitting

- Verify that your issue is not being currently addressed by other issues or pull requests
- If you are submitting an algorithm or feature request please verify that the algorithm fulfills our new algorithm requirements
- If you are submitting a bug report we strongly encourage you to follow the guidelines in How to make a good bug report

How to make a good bug report

When you submit an issue to Github please do your best to follow these guidelines This will make it a lot easier to provide you with good feedback

2408 Chapter 7 Developer’s Guide

scikitlearn user guide Release 0213

- The ideal bug report contains a short reproducible code snippet this way anyone can try to reproduce the bug easily see this for more details If your snippet is longer than around 50 lines please link to a gist or a github repo
- If not feasible to include a reproducible snippet please be specific about what estimators and/or functions are involved and the shape of the data
- If an exception is raised please provide the full traceback
- Please include your operating system type and version number as well as your Python scikitlearn numpy and scipy versions This information can be found by running the following code snippet

```
import sklearn
sklearn.show_versions
```

Note This utility function is only available in scikitlearn v0.20 For previous versions one has to explicitly run

```
import platform
print(platform.platform)
import sys
print(sys.version)
import numpy
print(numpy.__version__)
import scipy
print(scipy.__version__)
import sklearn
print(sklearn.__version__)
```

- Please ensure all code snippets and error messages are formatted in appropriate code blocks See Creating and highlighting code blocks for more details

7.1.3 Contributing code

Note To avoid duplicating work it is highly advised that you search through the issue tracker and the PR list If in doubt about duplicated work or if you want to work on a nontrivial feature it's recommended to first open an issue in the issue tracker to get some feedbacks from core developers

How to contribute

The preferred way to contribute to scikitlearn is to fork the main repository on GitHub then submit a "pull request" PR

- 1 Create an account on GitHub if you do not already have one
- 2 Fork the project repository click on the 'Fork' button near the top of the page This creates a copy of the code under your account on the GitHub user account For more details on how to fork a repository see this guide
- 3 Clone your fork of the scikitlearn repo from your GitHub account to your local disk

```
git clone git@github.com:YourLogin/scikitlearn.git
```

```
cd scikitlearn
```

- 4 Install the development dependencies

```
pip install cython pytest flake8
```

- 5 Install scikitlearn in editable mode

7.1.4 Contributing 2409

scikitlearn user guide Release 0213

pip install editable

for more details about advanced installation see the Building from source section

6 Add the upstream remote This saves a reference to the main scikitlearn repository which you can use to keep your repository synchronized with the latest changes

git remote add upstream <https://github.com/scikitlearn/scikitlearn.git>

7 Fetch the upstream and then create a branch to hold your development changes

git fetch upstream

git checkout -b myfeature upstream/master

and start making changes Always use a feature branch It's good practice to never work on the master branch

8 Develop the feature on your feature branch on your computer using Git to do the version control When you're done editing add changed files using git add and then git commit files

git add modified_files

git commit

to record your changes in Git then push the changes to your GitHub account with

git push -u origin myfeature

9 Follow these instructions to create a pull request from your fork This will send an email to the committers You may want to consider sending an email to the mailing list for more visibility

Note If you are modifying a Cython module you have to rerun step 5 after modifications and before testing them

It is often helpful to keep your local branch synchronized with the latest changes of the main scikitlearn repository

git fetch upstream

git merge upstream/master

Subsequently you might need to solve the conflicts You can refer to the Git documentation related to resolving merge conflict using the command line

Learning git

The Git documentation and <http://try.github.io> are excellent resources to get started with git and understanding all of the commands shown here

Pull request checklist

Before a PR can be merged it needs to be approved by two core developers Please prefix the title of your pull request with MRG if the contribution is complete and should be subjected to a detailed review An incomplete contribution

- where you expect to do more work before receiving a full review - should be prefixed WIP to indicate a work in progress and changed to MRG when it matures WIPs may be useful to indicate you are working on something to avoid duplicated work request broad review of functionality or API (Application Programming Interface) or seek collaborators WIPs often benefit from the inclusion of a task list in the PR description

2410 Chapter 7 Developer's Guide

scikitlearn user guide Release 0213

In order to ease the reviewing process we recommend that your contribution complies with the following rules before marking a PR as MRG. The bolded ones are especially important

1 Give your pull request a helpful title that summarises what your contribution does. This title will often become the commit message once merged so it should summarise your contribution for posterity. In some cases “Fix ISSUE TITLE” is enough. “Fix ISSUE NUMBER” is never a good title.

2 Make sure your code passes the tests. The whole test suite can be run with `pytest` but it is usually not recommended since it takes a long time. It is often enough to only run the test related to your changes for example if you changed something in `sklearn.linear_model.logistic.py` running the following commands will usually be enough

- `pytest sklearn.linear_model.logistic.py` to make sure the doctest examples are correct
- `pytest sklearn.linear_model.test.test_logistic.py` to run the tests specific to the file

- `pytest sklearn.linear_model` to test the whole Generalized Linear Models module
- `pytest sklearn.docstring_util` to make sure the user guide examples are correct
- `pytest sklearn.test.test_common.py` k LogisticRegression to run all our estimator checks specifically for LogisticRegression if that’s the estimator you changed

There may be other failing tests but they will be caught by the CI so you don’t need to run the whole test suite locally. You can read more in Testing and improving test coverage

3 Make sure your code is properly commented and documented and make sure the documentation renders properly. To build the documentation please refer to our Documentation guidelines. The CI will also build the docs please refer to Generated documentation on CircleCI

4 Tests are necessary for enhancements to be accepted. Bugfixes or new features should be provided with non regression tests. These tests verify the correct behavior of the fix or feature. In this manner further modifications on the code base are granted to be consistent with the desired behavior. In the case of bug fixes at the time of the PR the non regression tests should fail for the code base in the master branch and pass for the PR code

5 Make sure that your PR does not add PEP8 violations. On a Unixlike system you can run `make flake8diff` `flake8 path/to/file` would work for any system but please avoid reformatting parts of the file that your pull request doesn’t change as it distracts from code review

6 Follow the coding guidelines see below

7 When applicable use the validation tools and scripts in the `sklearn.utils` submodule. A list of utility routines available for developers can be found in the Utilities for Developers page

8 Often pull requests resolve one or more other issues or pull requests. If merging your pull request means that some other issues/PRs should be closed you should use keywords to create link to them eg `Fixes 1234` multiple issues/PRs are allowed as long as each one is preceded by a keyword. Upon merging those issues/PRs will automatically be closed by GitHub. If your pull request is simply related to some other issues/PRs create a link to them without using the keywords eg `See also 1234`

9 PRs should often substantiate the change through benchmarks of performance and efficiency or through examples of usage. Examples also illustrate the features and intricacies of the library to users. Have a look at other examples in the examples directory for reference. Examples should demonstrate why the new functionality is useful in practice and if possible compare it to other methods available in scikitlearn

10 New features often need to be illustrated with narrative documentation in the user guide with small code snippets. If relevant please also add references in the literature with PDF links when possible

11 The user guide should also include expected time and space complexity of the algorithm and scalability eg “this algorithm can scale to a large number of samples 100000 but does not scale in dimensionality n features is expected to be lower than 100”

71 Contributing 2411

scikitlearn user guide Release 0213

You can also check our Code Review Guidelines to get an idea of what reviewers will expect

You can check for common programming errors with the following tools

- Code with a good unittest coverage at least 80 better 100 check with
pip install pytest pytestcov
pytest cov sklearn pathtotestsforpackage

see also Testing and improving test coverage

Bonus points for contributions that include a performance analysis with a benchmark script and profiling output please report on the mailing list or on the GitHub issue

Also check out the How to optimize for speed guide for more details on profiling and Cython optimizations

Note The current state of the scikitlearn code base is not compliant with all of those guidelines but we expect that

enforcing those constraints on all new contributions will get the overall code base quality in the right direction

Note For two very well documented and more detailed guides on development workflow please pay a visit to the

Scipy Development Workflow and the Astropy Workflow for Developers sections

Continuous Integration CI

- Azure pipelines are used for testing scikitlearn on Linux Mac and Windows with different dependencies and settings

- CircleCI is used to build the docs for viewing for linting with flake8 and for testing with PyPy on Linux

Please note that if one of the following markers appear in the latest commit message the following actions are taken

Commit Message

MarkerAction Taken by CI

scipydev Add a Travis build with our dependencies numpy scipy etc develop
ment builds

ci skip CI is skipped completely

doc skip Docs are not built

doc quick Docs built but excludes example gallery plots

doc build Docs built including example gallery plots

Stalled pull requests

As contributing a feature can be a lengthy process some pull requests appear inactive but unfinished In such a case taking them over is a great service for the project

A good etiquette to take over is

- Determine if a PR is stalled

-A pull request may have the label “stalled” or “help wanted” if we have already identified it as a candidate for other contributors

2412 Chapter 7 Developer’s Guide

scikitlearn user guide Release 0213

-To decide whether an inactive PR is stalled ask the contributor if she/he plans to continue working on the PR in the near future Failure to respond within 2 weeks with an activity that moves the PR forward suggests that the PR is stalled and will result in tagging that PR with “help wanted”

Note that if a PR has received earlier comments on the contribution that have had no reply in a month it is safe to assume that the PR is stalled and to shorten the wait time to one day

After a sprint followup for unmerged PRs opened during sprint will be communicated to participants at the sprint and those PRs will be tagged “sprint” PRs tagged with “sprint” can be reassigned or declared stalled by sprint leaders

•Taking over a stalled PR To take over a PR it is important to comment on the stalled PR that you are taking over and to link from the new PR to the old one The new PR should be created by pulling from the old one

Issues for New Contributors

New contributors should look for the following tags when looking for issues We strongly recommend that new contributors tackle “easy” issues first this helps the contributor become familiar with the contribution workflow and for the core devs to become acquainted with the contributor besides which we frequently underestimate how easy an issue is to solve

good first issue tag

A great way to start contributing to scikitlearn is to pick an item from the list of good first issues in the issue tracker Resolving these issues allow you to start contributing to the project without much prior knowledge If you have already contributed to scikitlearn you should look at Easy issues instead

Easy tag

If you have already contributed to scikitlearn another great way to contribute to scikitlearn is to pick an item from the list of Easy issues in the issue tracker Your assistance in this area will be greatly appreciated by the more experienced developers as it helps free up their time to concentrate on other issues

help wanted tag

We often use the help wanted tag to mark issues regardless of difficulty Additionally we use the help wanted tag to mark Pull Requests which have been abandoned by their original contributor and are available for someone to pick up where the original contributor left off The list of issues with the help wanted tag can be found here

Note that not all issues which need contributors will have this tag

714 Documentation

We are glad to accept any sort of documentation function docstrings reStructuredText documents like this one tutorials etc reStructuredText documents live in the source code repository under the doc directory

You can edit the documentation using any text editor and then generate the HTML output by typing make from the doc directory Alternatively make html may be used to generate the documentation with the example gallery which takes quite some time The resulting HTML files will be placed in buildhtmlstable and are viewable in a web browser

71 Contributing 2413

scikitlearn user guide Release 0213

Building the documentation

First make sure you have properly installed the development version

Building the documentation requires installing some additional packages

pip install sphinx sphinxgallery numpydoc matplotlib Pillow pandas scikitimage

To build the documentation you need to be in the doc folder

cd doc

In the vast majority of cases you only need to generate the full web site without the example gallery

make

The documentation will be generated in the buildhtmlstable directory To also generate the example gallery

you can use

make html

This will run all the examples which takes a while If you only want to generate a few examples you can use

EXAMPLESPATTERNyourregexgoeshere make html

This is particularly useful if you are modifying a few examples

Set the environment variable NOMATHJAX1 if you intend to view the documentation in an offline setting

To build the PDF manual run

make latexpdf

Warning Sphinx version

While we do our best to have the documentation build under as many versions of Sphinx as possible the different

versions tend to behave slightly differently To get the best results you should use the same version as the one we

used on CircleCI Look at this github search to know the exact version

Guidelines for writing documentation

It is important to keep a good compromise between mathematical and algorithmic details and give intuition to the

reader on what the algorithm does

Basically to elaborate on the above it is best to always start with a small paragraph with a handwaving explanation of

what the method does to the data Then it is very helpful to point out why the feature is useful and when it should be

used the latter also including “big O” \mathcal{O} complexities of the algorithm as opposed to just rules of thumb as

the latter can be very machinedependent If those complexities are not available then rules of thumb may be provided

instead

Secondly a generated figure from an example as mentioned in the previous paragraph should then be included to

further provide some intuition

Next one or two small code examples to show its use can be added

2414 Chapter 7 Developer’s Guide

scikitlearn user guide Release 0213

Next any math and equations followed by references can be added to further the documentation Not starting the documentation with the maths makes it more friendly towards users that are just interested in what the feature will do as opposed to how it works “under the hood”

Finally follow the formatting rules below to make it consistently good

- Add “See also” in docstrings for related classesfunctions
- “See also” in docstrings should be one line per reference with a colon and an explanation for example
See also

SelectKBest Select features based on the k highest scores

SelectFpr Select features based on a false positive rate test

- For unwritten formatting rules try to follow existing good works
- For “References” in docstrings see the Silhouette Coefficient sklearnmetrics

silhouettescore

- When editing reStructuredText rst files try to keep line length under 80 characters when possible exceptions include links and tables

Generated documentation on CircleCI

When you change the documentation in a pull request CircleCI automatically builds it To view the documentation generated by CircleCI

- navigate to the bottom of your pull request page to see the CI statuses You may need to click on “Show all checks” to see all the CI statuses
- click on the CircleCI status with “doc” in the title
- addartifacts at the end of the URL Note you need to wait for the CircleCI build to finish before being able to look at the artifacts
- once the artifacts are visible navigate to docchangedhtml to see a list of documentation pages that are likely to be affected by your pull request Navigate to docindexhtml to see the full generated html documentation

If you often need to look at the documentation generated by CircleCI eg when reviewing pull requests you may find this tip very handy

715 Testing and improving test coverage

Highquality unit testing is a cornerstone of the scikitlearn development process For this purpose we use the pytest package The tests are functions appropriately named located in tests subdirectories that check the validity of the algorithms and the different options of the code

The full scikitlearn tests can be run using ‘make’ in the root folder Alternatively running ‘pytest’ in a folder will run all the tests of the corresponding subpackages

We expect code coverage of new features to be at least around 90

For guidelines on how to use pytest efficiently see the Useful pytest aliases and flags

71 Contributing 2415

scikitlearn user guide Release 0213

Writing matplotlib related tests

Test fixtures ensure that a set of tests will be executing with the appropriate initialization and cleanup The scikitlearn test suite implements a fixture which can be used with matplotlib
pyplot The pyplot fixture should be used when a test function is dealing with matplotlib matplotlib is a soft dependency and is not required This fixture is in charge of skipping the tests if matplotlib is not installed In addition figures created during the tests will be automatically closed once the test function has been executed

To use this fixture in a test function one needs to pass it as an argument

```
def test_requiring_mpl_fixture(pyplot):  
    you can now safely use matplotlib
```

Workflow to improve test coverage

To test code coverage you need to install the coverage package in addition to pytest

1 Run 'make testcoverage' The output lists for each file the line numbers that are not tested

2 Find a low hanging fruit looking at which lines are not tested write or adapt a test specifically for these

lines

3 Loop

Issue Tracker Tags

All issues and pull requests on the GitHub issue tracker should have at least one of the following tags

Bug Crash Something is happening that clearly shouldn't happen Wrong results as well as unexpected

errors from estimators go here

Cleanup Enhancement Improving performance usability consistency

Documentation Missing incorrect or substandard documentations and examples

New Feature Feature requests and pull requests implementing a new feature

There are four other tags to help new contributors

good first issue This issue is ideal for a first contribution to scikitlearn Ask for help if the formulation

is unclear If you have already contributed to scikitlearn look at Easy issues instead

Easy This issue can be tackled without much prior experience

Moderate Might need some knowledge of machine learning or the package but is still approachable for someone new to the project

help wanted This tag marks an issue which currently lacks a contributor or a PR that needs another contributor to take over the work These issues can range in difficulty and may not be approachable for new contributors Note that not all issues which need contributors will have this tag

716 Coding guidelines

The following are some guidelines on how new code should be written Of course there are special cases and there will be exceptions to these rules However following these rules when submitting new code makes the review easier so new code can be integrated in less time

scikitlearn user guide Release 0.21.3

Uniformly formatted code makes it easier to share code ownership The scikitlearn project tries to closely follow the official Python guidelines detailed in PEP8 that detail how code should be formatted and indented Please read it and follow it

In addition we add the following guidelines

- Use underscores to separate words in non class names `nsamples` rather than `nsamples`
- Avoid multiple statements on one line Prefer a line return after a control flow statement `ifor`
- Use relative imports for references inside scikitlearn
- Unit tests are an exception to the previous rule they should use absolute imports exactly as client code would

A corollary is that if `sklearnfoo` exports a class or function that is implemented in `sklearnfoobar` `baz` the test should import it from `sklearnfoo`

•Please don't use `importin` any case It is considered harmful by the official Python recommendations It makes the code harder to read as the origin of symbols is no longer explicitly referenced but most important it prevents using a static analysis tool like `pyflakes` to automatically find bugs in scikitlearn

- Use the `numpy` docstring standard in all your docstrings

A good example of code that we like can be found [here](#)

Input validation

The module `sklearnutils` contains various functions for doing input validation and conversion Sometimes `np.asarray` suffices for validation do not use `np.asarray` or `np.atleast2d` since those let NumPy's `np.matrix` through which has a different API (Application Programming Interface) eg means dot product on `np.matrix` but Hadamard product on `np.ndarray`

In other cases be sure to call `checkarray` on any arraylike argument passed to a scikitlearn API (Application Programming Interface) function The exact parameters to use depends mainly on whether and which `scipy` sparse matrices must be accepted

For more information refer to the [Utilities for Developers](#) page

Random Numbers

If your code depends on a random number generator do not use `numpy.random.random` or similar routines

To ensure repeatability in error checking the routine should accept a keyword `randomstate` and use this to construct a `numpy.random.RandomState` object See `sklearnutils.checkrandomstate` in [Utilities for Developers](#)

Here's a simple example of code using some of the above guidelines

```
from sklearnutils import checkarray checkrandomstate
defchoose_random_sample(X, randomstate=0)
```

Choose a random point from `X`

Parameters

`X` arraylike shape `(nsamples, nfeatures)`
array representing the data
`randomstate` `RandomState` or an int seed 0 by default
A random number generator instance to define the state of the random permutations generator

x numpy array shape nfeatures
A random point selected from X

X checkarrayX
randomstate checkrandomstaterandomstate
i randomstaterandintXshape0
returnXi

If you use randomness in an estimator instead of a freestanding function some additional guidelines apply
First off the estimator should take a randomstate argument to its init with a default value of
None It should store that argument's value unmodified in an attribute randomstate fit can call
checkrandomstate on that attribute to get an actual random number generator If for some reason ran
domness is needed after fit the RNG should be stored in an attribute randomstate The following example
should make this clear
class GaussianNoise BaseEstimator TransformerMixin
This estimator ignores its input and returns random Gaussian noise
It also does not adhere to all scikitlearn conventions
but showcases how to handle randomness

```
definitself ncomponents100 randomstate None
selfrandomstate randomstate
    the arguments are ignored anyway so we make them optional
deffitself X None yNone
selfrandomstate checkrandomstateselfrandomstate
deftransformself X
nsamples Xshape0
returnselfrandomstaterandnnsamples ncomponents
```

The reason for this setup is reproducibility when an estimator is fit twice to the same data it should produce an
identical model both times hence the validation in fit notinit
Deprecation

If any publicly accessible method function attribute or parameter is renamed we still support the old one for two
releases and issue a deprecation warning when it is calledpassedaccessed Eg if the function zeroone is re
named tozerooneloss we add the decorator deprecated fromsklearnutils tozeroone and call
zerooneloss from that function
from utils import deprecated
defzeroonelossytrue ypred normalize True
 actual implementation
pass
deprecated Function zeroone was renamed to zerooneloss
in version 013 and will be removed in release 015
Default behavior is changed from normalizeFalse to
normalizeTrue

scikitlearn user guide Release 0213

```
defzerooneytrue ypred normalize False
returnzeroonelossytrue ypred normalize
```

If an attribute is to be deprecated use the decorator deprecated on a property Please note that the property decorator should be placed before the deprecated decorator for the docstrings to be rendered properly Eg

renaming an attribute labels toclasses can be done as
deprecated Attribute labels was deprecated in version 013 and
will be removed in 015 Use classes instead

```
property
deflabelsself
returnselfclasses
```

If a parameter has to be deprecated use DeprecationWarning appropriately In the following example k is deprecated and renamed to nclusters

```
import warnings
defexamplefunctionnclusters8 knotused
ifk notused
```

warningswarnk was renamed to nclusters in version 013 and
will be removed in 015 DeprecationWarning
nclusters k

When the change is in a class we validate and raise warning in fit

```
import warnings
class ExampleEstimator BaseEstimator
definitself nclusters8 knotused
selfnclusters nclusters
```

```
selfk k
deffitself X y
ifselfk notused
warningswarnk was renamed to nclusters in version 013 and
will be removed in 015 DeprecationWarning
selfnclusters selfk
else
selfnclusters selfnclusters
```

As in these examples the warning message should always give both the version in which the deprecation happened and the version in which the old behavior will be removed If the deprecation happened in version 0xdev the message should say deprecation occurred in version 0x and the removal will be in 0x2 so that users will have enough time to adapt their code to the new behaviour For example if the deprecation happened in version 018dev the message should say it happened in version 018 and the old behavior will be removed in version 020

In addition a deprecation note should be added in the docstring recalling the same information as the deprecation warning as explained above Use the deprecated directive
deprecated 013

k was renamed to nclusters in version 013 and will be removed
in 015

What's more a deprecation requires a test which ensures that the warning is raised in relevant cases but not in other cases The warning should be caught in all other tests using eg pytestmarkfilterwarnings and there should be no warning in the examples

71 Contributing 2419

scikitlearn user guide Release 0213

Change the default value of a parameter

If the default value of a parameter needs to be changed please replace the default value with a specific value eg warn and raiseFutureWarning when users are using the default value In the following example we change the default value of nclusters from 5 to 10 current version is 020

```
import warnings
defexamplefunctionnclusterswarn
ifnclusters warn
warningswarnThe default value of nclusters will change from
5 to 10 in 022 FutureWarning
nclusters 5
```

When the change is in a class we validate and raise warning in fit

```
import warnings
class ExampleEstimator
definitself nclusterswarn
selfnclusters nclusters
deffitself X y
ifselfnclusters warn
warningswarnThe default value of nclusters will change from
5 to 10 in 022 FutureWarning
selfnclusters 5
```

Similar to deprecations the warning message should always give both the version in which the change happened and the version in which the old behavior will be removed The docstring needs to be updated accordingly We need a test which ensures that the warning is raised in relevant cases but not in other cases The warning should be caught in all other tests using eg pytestmarkfilterwarnings and there should be no warning in the examples

Python versions supported

Since scikitlearn 021 only Python 35 and newer is supported

717 Code Review Guidelines

Reviewing code contributed to the project as PRs is a crucial component of scikitlearn development We encourage anyone to start reviewing code of other developers The code review process is often highly educational for everybody involved This is particularly appropriate if it is a feature you would like to use and so can respond critically about whether the PR meets your needs While each pull request needs to be signed off by two core developers you can speed up this process by providing your feedback

Here are a few important aspects that need to be covered in any code review from highlevel questions to a more detailed checklist

- Do we want this in the library Is it likely to be used Do you as a scikitlearn user like the change and intend to use it Is it in the scope of scikitlearn Will the cost of maintaining a new feature be worth its benefits
- Is the code consistent with the API (Application Programming Interface) of scikitlearn Are public functionsclassesparameters well named and intuitively designed
- Are all public functionsclasses and their parameters return types and stored attributes named according to scikitlearn conventions and documented clearly

scikitlearn user guide Release 0213

- Is any new functionality described in the userguide and illustrated with examples
- Is every public functionclass tested Are a reasonable set of parameters their values value types and combinations tested Do the tests validate that the code is correct ie doing what the documentation says it does If the change is a bugfix is a nonregression test included Look at this to get started with testing in Python
- Do the tests pass in the continuous integration build If appropriate help the contributor understand why tests failed
- Do the tests cover every line of code see the coverage report in the build log If not are the lines missing coverage good exceptions
- Is the code easy to read and low on redundancy Should variable names be improved for clarity or consistency Should comments be added Should comments be removed as unhelpful or extraneous
- Could the code easily be rewritten to run much more efficiently for relevant settings
- Is the code backwards compatible with previous versions or is a deprecation cycle necessary
- Will the new code add any dependencies on other libraries this is unlikely to be accepted
- Does the documentation render properly see the Documentation section for more details and are the plots instructive

Standard replies for reviewing includes some frequent comments that reviewers may make

718 APIs of scikitlearn objects

To have a uniform API (Application Programming Interface) we try to have a common basic API (Application Programming Interface) for all the objects of framework code we try to adopt simple conventions and limit to a minimum the number of methods an object must implement

Elements of the scikitlearn API (Application Programming Interface) are described more definitively in the Glossary of Common Terms and API (Application Programming Interface)

Different objects

The main objects in scikitlearn are one class can implement multiple interfaces

Estimator The base object implements a fit method to learn from data either

estimator estimatorfitdata targets

or

estimator estimatorfitdata

Predictor For supervised learning or some unsupervised problems implements

prediction predictorpredictdata

Classification algorithms usually also offer a way to quantify certainty of a prediction either using

decisionfunction orpredictproba

probability predictorpredictprobadata

Transformer For filtering or modifying the data in a supervised or unsupervised way implements

newdata transformertransformdata

71 Contributing 2421

scikitlearn user guide Release 0213

When fitting and transforming can be performed much more efficiently together than separately implements

newdata transformerfittransformdata

Model A model that can give a goodness of fit measure or a likelihood of unseen data implements higher is better

score modelscoredata

Estimators

The API (Application Programming Interface) has one predominant object the estimator A estimator is an object that fits a model based on some training data and is capable of inferring some properties on new data It can be for instance a classifier or a regressor All

estimators implement the fit method

estimatorfitX y

All builtin estimators also have a setparams method which sets dataindependent parameters overriding previous parameter values passed to init

All estimators in the main scikitlearn codebase should inherit from sklearnbaseBaseEstimator

Instantiation

This concerns the creation of an object The object’s init method might accept constants as arguments that determine the estimator’s behavior like the C constant in SVMs It should not however take the actual training data as an argument as this is left to the fit method

clf2 SVCC23

clf3 SVC1 2 2 3 1 1 WRONG

The arguments accepted by init should all be keyword arguments with a default value In other words a user should be able to instantiate an estimator without passing any arguments to it The arguments should all correspond to hyperparameters describing the model or the optimisation problem the estimator tries to solve These initial arguments or parameters are always remembered by the estimator Also note that they should not be documented under the “Attributes” section but rather under the “Parameters” section for that estimator

In addition every keyword argument accepted by init should correspond to an attribute on the instance

Scikitlearn relies on this to find the relevant attributes to set on an estimator when doing model selection

To summarize an init should look like

definitself param11 param22

selfparam1 param1

selfparam2 param2

There should be no logic not even input validation and the parameters should not be changed The corresponding logic should be put where the parameters are used typically in fit The following is wrong

definitself param11 param22 param33

WRONG parameters should not be modified

ifparam1 1

param2 1

selfparam1 param1

2422 Chapter 7 Developer’s Guide

scikitlearn user guide Release 0213

WRONG the objects attributes should have exactly the name of the argument in the constructor

selfparam3 param2

The reason for postponing the validation is that the same validation would have to be performed in setparams which is used in algorithms like GridSearchCV

Fitting

The next thing you will probably want to do is to estimate some parameters in the model This is implemented in the fit method

Thefit method takes the training data as arguments which can be one array in the case of unsupervised learning or two arrays in the case of supervised learning

Note that the model is fitted using X and y but the object holds no reference to X and y There are however some exceptions to this as in the case of precomputed kernels where this data must be stored for use by the predict method

Parameters

X arraylike shape nsamples nfeatures

y array shape nsamples

kwargs optional datadependent parameters

Xshape0 should be the same as yshape0 If this requisite is not met an exception of type ValueError should be raised

ymight be ignored in the case of unsupervised learning However to make it possible to use the estimator as part of a pipeline that can mix both supervised and unsupervised transformers even unsupervised estimators need to accept

ayNone keyword argument in the second position that is just ignored by the estimator For the same reason

fitpredict fittransform score andpartialfit methods need to accept a yargument in the second place if they are implemented

The method should return the object self This pattern is useful to be able to implement quick one liners in an IPython session such as

ypredicted SVCC100fitXtrain ytrainpredictXtest

Depending on the nature of the algorithm fit can sometimes also accept additional keywords arguments However any parameter that can have a value assigned prior to having access to the data should be an init keyword

argument fit parameters should be restricted to directly data dependent variables For instance a Gram matrix or an affinity matrix which are precomputed from the data matrix Xare data dependent A tolerance stopping criterion

tol is not directly data dependent although the optimal value according to some scoring function probably is

Whenfit is called any previous call to fit should be ignored In general calling estimatorfitX1 and

thenestimatorfitX2 should be the same as only calling estimatorfitX2 However this may not

be true in practice when fit depends on some random process see randomstate Another exception to this rule is

when the hyperparameter warmstart is set toTrue for estimators that support it warmstartTrue means

that the previous state of the trainable parameters of the estimator are reused instead of using the default initialization strategy

Estimated Attributes

Attributes that have been estimated from the data must always have a name ending with trailing underscore for example the coefficients of some regression estimator would be stored in a coef attribute after fit has been called

71 Contributing 2423

scikitlearn user guide Release 0213

The estimated attributes are expected to be overridden when you call fit a second time

Optional Arguments

In iterative algorithms the number of iterations should be specified by an integer called niter

Pairwise Attributes

An estimator that accept X of shape samples n samples and defines a pairwise property equal to True allows for crossvalidation of the dataset eg when X is a precomputed kernel matrix Specifically the pairwise property is used by utilmetaestimatorssafesplit to slice rows and columns

719 Rolling your own estimator

If you want to implement a new estimator that is scikitlearn compatible whether it is just for you or for contributing it to scikitlearn there are several internals of scikitlearn that you should be aware of in addition to the scikitlearn API (Application Programming Interface) outlined above You can check whether your estimator adheres to the scikitlearn interface and standards by running

```
from sklearn.util.estimator_checks import check_estimator
from sklearn.svm import LinearSVC
check_estimator(LinearSVC)  # passes
```

The main motivation to make a class compatible to the scikitlearn estimator interface might be that you want to use it together with model evaluation and selection tools such as modelselectionGridSearchCV and pipelinePipeline

Before detailing the required interface below we describe two ways to achieve the correct interface more easily Project template

We provide a project template which helps in the creation of Python packages containing scikitlearn compatible estimators It provides

- an initial git repository with Python package directory structure
- a template of a scikitlearn estimator
- an initial test suite including use of check_estimator
- directory structures and scripts to compile documentation and example galleries
- scripts to manage continuous integration testing on Linux and Windows
- instructions from getting started to publishing on PyPi

BaseEstimator and mixins

We tend to use “duck typing” so building an estimator which follows the API (Application Programming Interface) suffices for compatibility without needing to inherit from or even import any scikitlearn classes

2424 Chapter 7 Developer’s Guide

scikitlearn user guide Release 0213

However if a dependency on scikitlearn is acceptable in your code you can prevent a lot of boilerplate code by deriving a class from BaseEstimator and optionally the mixin classes in sklearnbase For example below is a custom classifier with more examples included in the scikitlearncontrib project template

```
import numpy as np
from sklearnbase import BaseEstimator ClassifierMixin
from sklearnutilsvalidation import checkXy checkarray checkisfitted
from sklearnutilsmulticlass import uniquelabels
from sklearnmetrics import euclidean_distances
class TemplateClassifier BaseEstimator ClassifierMixin
```

```
def initself demoparamdemo
selfdemoparam demoparam
```

```
def fitself X y

    Check that X and y have correct shape
    X y checkXyX y
    Store the classes seen during fit
    selfclasses uniquelabelsy
```

```
selfX X
selfy y
    Return the classifier
return self
```

```
def predictself X

    Check is fit had been called
    checkisfittedself X y
```

```
    Input validation
    X checkarrayX
```

```
closest npargmineuclidean_distancesX selfX axis1
return selfyclosest
```

getparams and setparams

All scikitlearn estimators have getparams andsetparams functions The getparams function takes no arguments and returns a dict of the init parameters of the estimator together with their values It must take one keyword argument deep which receives a boolean value that determines whether the method should return the parameters of subestimators for most estimators this can be ignored The default value for deep should be true Thesetparams on the other hand takes as input a dict of the form parameter value and sets the parameter of the estimator using this dict Return value must be estimator itself

While thegetparams mechanism is not essential see Cloning below the setparams function is necessary as it is used to set parameters during grid searches

The easiest way to implement these functions and to get a sensible repr method is to inherit from sklearn baseBaseEstimator If you do not want to make your code dependent on scikitlearn the easiest way to implement the interface is

```
defgetparamsself deep True
```

```
    suppose this estimator has parameters alpha and recursive
returnalpha selfalpha recursive selfrecursive
```

```
scikitlearn user guide Release 0213
defsetparamself parameters
forparameter value inparametersitems
setattrself parameter value
returnself
```

Parameters and init

AsmodelselectionGridSearchCV usessetparams to apply parameter setting to estimators it is essential that calling setparams has the same effect as setting parameters using the init method The easiest and recommended way to accomplish this is to not do any parameter validation in init All logic behind estimator parameters like translating string arguments into functions should be done in fit Also it is expected that parameters with trailing arenot to be set inside the init method All and only the public attributes set by fit have a trailing As a result the existence of parameters with trailing is used to check if the estimator has been fitted

Cloning

For use with the modelselection module an estimator must support the baseclone function to replicate an estimator This can be done by providing a getparams method If getparams is present then cloneestimator will be an instance of typeestimator on whichsetparams has been called with clones of the result of estimatorgetparams Objects that do not provide this method will be deepcopied using the Python standard function copydeepcopy ifsafeFalse is passed to clone

Pipeline compatibility

For an estimator to be usable together with pipelinePipeline in any but the last step it needs to provide a fit orfittransform function To be able to evaluate the pipeline on any data but the training set it also needs to provide atransform function There are no special requirements for the last step in a pipeline except that it has a fit function All fit andfittransform functions must take arguments X y even if y is not used Similarly forscore to be usable the last step of the pipeline needs to have a score function that accepts an optional y

Estimator types

Some common functionality depends on the kind of estimator passed For example crossvalidation in modelselectionGridSearchCV andmodelselectioncrossvalscore defaults to being stratified when used on a classifier but not otherwise Similarly scorers for average precision that take a continuous prediction need to call decisionfunction for classifiers but predict for regressors This distinction between classifiers and regressors is implemented using the estimatortype attribute which takes a string value It should be classifier for classifiers and regressor for regressors and clusterer for clustering methods to work as expected Inheriting from ClassifierMixin RegressorMixin orClusterMixin will set the attribute automatically When a metaestimator needs to distinguish among estimator types instead of checking estimatortype directly helpers like baseisclassifier should be used

Specific models

Classifiers should accept ytarget arguments to fit that are sequences lists arrays of either strings or integers They should not assume that the class labels are a contiguous range of integers instead they should store a list of

scikitlearn user guide Release 0213

classes in a classes attribute or property The order of class labels in this attribute should match the order in which predictproba predictlogproba anddecisionfunction return their values The easiest way to achieve this is to put

```
self.classes_ = np.unique(y)
return inverse_transform(self.classes_, y)
```

infit This returns a new ythat contains class indexes rather than labels in the range 0 nclasses

A classifier’s predict method should return arrays containing class labels from classes In a classifier that implements decisionfunction this can be achieved with

```
def predict(self, X):
```

```
    D = self.decision_function(X)
```

```
    return self.classes_[np.argmax(D, axis=1)]
```

In linear models coefficients are stored in an array called coef and the independent term is stored in intercept

sklearnlinearmodelbase contains a few base classes and mixins that implement common linear model

patterns

The sklearnutilsmulticlass module contains useful functions for working with multiclass and multilabel

problems

Estimator Tags

Warning The estimator tags are experimental and the API (Application Programming Interface) is subject to change

Scikitlearn introduced estimator tags in version 021 These are annotations of estimators that allow programmatic

inspection of their capabilities such as sparse matrix support supported output types and supported methods The

estimator tags are a dictionary returned by the method gettags These tags are used by the common tests and

thesklearnutilsestimatorcheckscheckestimator function to decide what tests to run and what

input data is appropriate Tags can depend on estimator parameters or even system architecture and can in general only

be determined at runtime

The default value of all tags except for Xtypes isFalse These are defined in the BaseEstimator class

The current set of estimator tags are

nondeterministic whether the estimator is not deterministic given a fixed randomstate

requirespositivedata unused for now whether the estimator requires positive X

novalidation whether the estimator skips inputvalidation This is only meant for stateless and dummy transformers

multioutput unused for now whether a regressor supports multitarget outputs or a classifier supports multiclass

multioutput

multilabel whether the estimator supports multilabel output

stateless whether the estimator needs access to data for fitting Even though an estimator is stateless it might still

need a call to fit for initialization

allownan whether the estimator supports data with missing values encoded as npNaN

poorscore whether the estimator fails to provide a “reasonable” testset score which currently for regression

is an R2 of 05 on a subset of the boston housing dataset and for classification an accuracy of 083 on

makeblobsnsamples300 randomstate0 These datasets and values are based on current

estimators in sklearn and might be replaced by something more systematic

multioutputonly whether estimator supports only multioutput classification or regression

71 Contributing 2427

scikitlearn user guide Release 0213

skiptest whether to skip common tests entirely Don't use this unless you have a very good reason
Xtypes Supported input types for X as list of strings Tests are currently only run if '2darray' is contained in the list
signifying that the estimator takes continuous 2d numpy arrays as input The default value is '2darray' Other
possible types are string sparse categorical dict 1dlabels and 2dlabels
The goal is that in the future the supported input type will determine the data used during testing in particular
forstring sparse andcategorical data For now the test for sparse data do not make use of
thesparse tag

To override the tags of a child class one must define the moretags method and return a dict with the desired
tags eg
class MyMultiOutputEstimator BaseEstimator
defmoretagsself
returnmultioutputonly True
nondeterministic True

In addition to the tags estimators also need to declare any nonoptional parameters to init in the
requiredparameters class attribute which is a list or tuple If requiredparameters is only
estimator orbaseestimator then the estimator will be instantiated with an instance of
LinearDiscriminantAnalysis orRidgeRegression if the estimator is a regressor in the tests The
choice of these two models is somewhat idiosyncratic but both should provide robust closedform solutions

7110 Reading the existing code base

Reading and digesting an existing code base is always a difficult exercise that takes time and experience to master
Even though we try to write simple code in general understanding the code can seem overwhelming at first given the
sheer size of the project Here is a list of tips that may help make this task easier and faster in no particular order

- Get acquainted with the APIs of scikitlearn objects understand what fitpredict transform etc are used for
- Before diving into reading the code of a function class go through the docstrings first and try to get an idea of
what each parameter attribute is doing It may also help to stop a minute and think how would I do this myself
if I had to

- The trickiest thing is often to identify which portions of the code are relevant and which are not In scikit
learn a lot of input checking is performed especially at the beginning of the fitmethods Sometimes
only a very small portion of the code is doing the actual job For example looking at the fit method
ofsklearnlinearmodelLinearRegression what you're looking for might just be the call the
scipylininalglsq but it is buried into multiple lines of input checking and the handling of different
kinds of parameters

- Due to the use of Inheritance some methods may be implemented in parent classes All estimators inherit
at least from BaseEstimator and from a Mixin class eg ClassifierMixin that enables default
behaviour depending on the nature of the estimator classifier regressor transformer etc
- Sometimes reading the tests for a given function will give you an idea of what its intended purpose is You can
usegit grep see below to find all the tests written for a function Most tests for a specific functionclass
are placed under the tests folder of the module
- You'll often see code looking like this out Parallel
delayedsomefunctionparam for param in someiterable This runs
somefunction in parallel using Joblib out is then an iterable containing the values returned by
somefunction for each call

- We use Cython to write fast code Cython code is located in pyx andpxd files Cython code has a more
Clike flavor we use pointers perform manual memory allocation etc Having some minimal experience in C

2428 Chapter 7 Developer's Guide

scikitlearn user guide Release 0213

C is pretty much mandatory here

- Master your tools

-With such a big project being efficient with your favorite editor or IDE goes a long way towards digesting the code base Being able to quickly jump or peek to a functionclassattribute definition helps a lot So does being able to quickly see where a given name is used in a file

-git also has some builtin killer features It is often useful to understand how a file changed over time using eggit blame manual This can also be done directly on GitHub git grep examples is also extremely useful to see every occurrence of a pattern eg a function call or a variable in the code base

72 Developers' Tips and Tricks

721 Productivity and sanitypreserving tips

In this section we gather some useful advice and tools that may increase your qualityoflife when reviewing pull requests running unit tests and so forth Some of these tricks consist of userscripts that require a browser extension such as TamperMonkey or GreaseMonkey to set up userscripts you must have one of these extensions installed enabled and running We provide userscripts as GitHub gists to install them click on the "Raw" button on the gist page

Viewing the rendered HTML documentation for a pull request

We use CircleCI to build the HTML documentation for every pull request To access that documentation instructions are provided in the documentation section of the contributor guide To save you a few clicks we provide a userscript that adds a button to every PR After installing the userscript navigate to any GitHub PR a new button labeled "See CircleCI doc for this PR" should appear in the topright area

Folding and unfolding outdated diffs on pull requests

GitHub hides discussions on PRs when the corresponding lines of code have been changed in the mean while This userscript provides a shortcut ControlAltP at the time of writing but look at the code to be sure to unfold all such hidden discussions at once so you can catch up

Checking out pull requests as remotetracking branches

In your local fork add to your gitconfig under theremote upstream heading the line

fetch refspull headrefsremotesupstreampr

You may then use git checkout prPRNUMBER to navigate to the code of the pullrequest with the given number Read more in this gist

Display code coverage in pull requests

To overlay the code coverage reports generated by the CodeCov continuous integration consider this browser extension The coverage of each line will be displayed as a color background behind the line number

72 Developers' Tips and Tricks 2429

scikitlearn user guide Release 0213

Useful pytest aliases and flags

The full test suite takes fairly long to run For faster iterations it is possibly to select a subset of tests using pytest selectors In particular one can run a single test based on its node ID

pytest v sklearnlinearmodelteststestlogisticpytestsparsify
or use the k pytest parameter to select tests based on their name For instance

pytest sklearnnteststestcommonpy v k LogisticRegression
will run all common tests for theLogisticRegression estimator

When a unit test fails the following tricks can make debugging easier

1 The command line argument pytest l instructs pytest to print the local variables when a failure occurs
2 The argument pytest pdb drops into the Python debugger on failure To instead drop into the rich IPython

debuggeripdb you may set up a shell alias to
pytest pdbcslIPythonterminaldebuggerTerminalPdb capture no

Otherpytest options that may become useful include

- xwhich exits on the first failed test
- lf to rerun the tests that failed on the previous run
- ff to rerun all previous tests running the ones that failed first
- sso that pytest does not capture the output of print statements
- tbshort ortbline to control the length of the logs

Since our continuous integration tests will error if DeprecationWarning orFutureWarning aren't prop
erly caught it is also recommended to run pytest along with the WerrorDeprecationWarning and

WerrorFutureWarning flags

Standard replies for reviewing

It may be helpful to store some of these in GitHub's saved replies for reviewing

Issue Usage questions

You're asking a usage question The issue tracker is mainly for bugs and new

↪features For usage questions it is recommended to try Stack Overflowhttps

↪stackoverflowcomquestiontaggedscikitlearn or the Mailing Listhttps

↪mailpythonorgmailmanlistinfooscikitlearn

Issue You're welcome to update the docs

Please feel free to offer a pull request updating the documentation if you feel

↪it could be improved

Issue Selfcontained example for bug

Please provide selfcontained example codehttpsstackoverflowcomhelpmcve

↪including imports and data if possible so that other contributors can just

↪run it and reproduce your issue Ideally your example code should be minimal

2430 Chapter 7 Developer's Guide

scikitlearn user guide Release 0213
Issue Software versions
To help diagnose your issue please paste the output of
py
import sklearn sklearnshowversions

Thanks
Issue Code blocks
Readability can be greatly improved if you format <https://help.github.com/articles/creating-and-highlighting-code-blocks> your code snippets and
complete error messages appropriately For example
python
print(something)

generates
python
print(something)

And
pytb
Traceback most recent call last
File stdin line 1 in module
ImportError No module named hello

generates
pytb
Traceback most recent call last
File stdin line 1 in module
ImportError No module named hello

You can edit your issue descriptions and comments at any time to improve
readability This helps maintainers a lot Thanks
IssueComment Linking to code
Friendly advice for claritys sake you can link to code like this <https://help.github.com/articles/creating-a-permanent-link-to-a-code-snippet>
IssueComment Linking to comments
Please use links to comments which make it a lot easier to see what you are
referring to rather than just linking to the issue See this <https://stackoverflow.com/questions/25163598/how-to-reference-a-specific-issue>
comment on github for more details
PRNEW Better description
Thanks for the pull request Please make the title of the PR descriptive so that
we can easily recall the issue it is resolving You should state what issue or
PR it fixes/resolves in the description see here <https://scikitlearn.org/dev/developers-contributing.html#contributing-pull-requests>
PRNEW Fix
72 Developers' Tips and Tricks 2431

scikitlearn user guide Release 0213

Please use `Fix issueNumber` in your PR description and you can do it more than once This way the associated issue gets closed automatically when the PR is merged For more details look at [this https://github.com/blog/1506-closing-issues-via-pull-requests](https://github.com/blog/1506-closing-issues-via-pull-requests)

PRNEW or Issue Maintenance cost

Every feature we include has a maintenance cost <https://scikitlearn.org/devfaq.html#why-are-you-so-selective-on-what-algorithms-you-include-in-scikitlearn>

Our maintainers are mostly volunteers For a new feature to be included we need evidence that it is often useful and ideally well established <https://scikitlearn.org/devfaq.html#what-are-the-inclusion-criteria-for-new-algorithms-in-the-literature-or-in-practice> That doesn't stop you implementing it for yourself and publishing it in a separate repository or even [scikitlearn-contrib https://scikitlearn-contrib.github.io](https://scikitlearn-contrib.github.io)

PRWIP What's needed before merge

Please clarify perhaps as a TODO list in the PR description what work you believe still needs to be done before it can be reviewed for merge When it is ready please prefix the PR title with MRG

PRWIP Regression test needed

Please add a nonregression test https://en.wikipedia.org/wiki/Nonregression_testing that would fail at master but pass in this PR

PRWIP PEP8

You have some PEP8 <https://www.python.org/dev/peps/pep-0008> violations whose details you can see in the Circle CI lint job It might be worth configuring your code editor to check for such errors on the fly so you can catch them before committing

PRMRG Patience

Before merging we generally require two core developers to agree that your pull request is desirable and ready Please be patient <https://scikitlearn.org/devfaq.html#why-is-my-pull-request-not-getting-any-attention> as we mostly rely on volunteered time from busy core developers You are also welcome to help us out with reviewing other PRs <https://scikitlearn.org/dev/developers-contributing.html#code-review-guidelines>

PRMRG Add to what's new

Please add an entry to the change log at `doc/whatsnew/vrst` Like the other entries there please reference this pull request with `pr` and credit yourself and other contributors if applicable with `user`

PR Don't change unrelated

Please do not change unrelated lines It makes your contribution harder to review and may introduce merge conflicts to other pull requests

2432 Chapter 7 Developer's Guide

scikitlearn user guide Release 0213

722 Debugging memory errors in Cython with valgrind

While pythonnumpy's builtin memory management is relatively robust it can lead to performance penalties for some routines For this reason much of the highperformance code in scikitlearn is written in cython This performance gain comes with a tradeoff however it is very easy for memory bugs to crop up in cython code especially in situations where that code relies heavily on pointer arithmetic

Memory errors can manifest themselves a number of ways The easiest ones to debug are often segmentation faults and related glibc errors Uninitialized variables can lead to unexpected behavior that is difficult to track down A very useful tool when debugging these sorts of errors is valgrind

Valgrind is a commandline tool that can trace memory errors in a variety of code Follow these steps

1 Install valgrind on your system

2 Download the python valgrind suppression file valgrindpythonsupp

3 Follow the directions in the READMEvalgrind file to customize your python suppressions If you don't you will have spurious output coming related to the python interpreter instead of your own code

4 Run valgrind as follows

valgrind v suppressionsvalgrindpythonsupp python mytestscriptpy

The result will be a list of all the memoryrelated errors which reference lines in the Ccode generated by cython from your pyx file If you examine the referenced lines in the c file you will see comments which indicate the corresponding location in your pyx source file Hopefully the output will give you clues as to the source of your memory error

For more information on valgrind and the array of options it has see the tutorials and documentation on the valgrind web site

73 Utilities for Developers

Scikitlearn contains a number of utilities to help with development These are located in sklearnutils and include tools in a number of categories All the following functions and classes are in the module sklearnutils Warning These utilities are meant to be used internally within the scikitlearn package They are not guaranteed to be stable between versions of scikitlearn Backports in particular will be removed as the scikitlearn dependencies evolve

731 Validation Tools

These are tools used to check and validate input When you write a function which accepts arrays matrices or sparse matrices as arguments the following should be used when applicable

- assertallfinite Throw an error if array contains NaNs or Infs
- asfloatarray convert input to an array of floats If a sparse matrix is passed a sparse matrix will be returned
- checkarray check that input is a 2D array raise error on sparse matrices Allowed sparse matrix formats can be given optionally as well as allowing 1D or Ndimensional arrays Calls assertallfinite by default

73 Utilities for Developers 2433

scikitlearn user guide Release 0213

- checkXy check that X and y have consistent length calls checkarray on X and columnor1d on y For multilabel classification or multitarget regression specify multioutputTrue in which case checkarray will be called on y
- indexable check that all input arrays have consistent length and can be sliced or indexed using safeindex This is used to validate input for crossvalidation
- validationcheckmemory checks that input is joblibMemory like which means that it can be converted into a sklearnutilsMemory instance typically a str denoting the cachedir or has the same interface

If your code relies on a random number generator it should never use functions like numpyrandomrandom ornumpyrandomnormal This approach can lead to repeatability issues in unit tests Instead a numpy randomRandomState object should be used which is built from a randomstate argument passed to the class or function The function checkrandomstate below can then be used to create a random number generator object

- checkrandomstate create anrandomRandomState object from a parameter randomstate
- Ifrandomstate isNone ornp.random then a randomlyinitialized RandomState object is re turned
- Ifrandomstate is an integer then it is used to seed a new RandomState object
- Ifrandomstate is aRandomState object then it is passed through

For example

```
from sklearnutils import checkrandomstate
randomstate 0
randomstate checkrandomstaterandomstate
randomstaterand4
array05488135 071518937 060276338 054488318
```

When developing your own scikitlearn compatible estimator the following helpers are available

- validationcheckisfitted check that the estimator has been fitted before calling transform predict or similar methods This helper allows to raise a standardized error message across estimator
- validationhasfitparameter check that a given parameter is supported in the fit method of a given estimator

732 Efficient Linear Algebra Array Operations

- extmathrandomizedrangefinder construct an orthonormal matrix whose range approximates the range of the input This is used in extmathrandomizedsvd below
- extmathrandomizedsvd compute the ktruncated randomized SVD This algorithm finds the exact truncated singular values decomposition using randomization to speed up the computations It is particularly fast on large matrices on which you wish to extract only a small number of components
- arrayfuncscholeskydelete used insklearnlinearmodelarspath Remove an item from a cholesky factorization
- arrayfunscminpos used insklearnlinearmodelleangle Find the minimum of the positive values within an array
- extmathfastlogdet efficiently compute the log of the determinant of a matrix
- extmathdensity efficiently compute the density of a sparse vector

scikitlearn user guide Release 0213

- extmathsafeparsedot dot product which will correctly handle scipysparse inputs If the inputs are dense it is equivalent to numpydot
- extmathweightedmode an extension of scipystatsmode which allows each item to have a real valued weight

- resample Resample arrays or sparse matrices in a consistent way used in shuffle below
- shuffle Shuffle arrays or sparse matrices in a consistent way Used in sklearnclusterkmeans

733 Efficient Random Sampling

- randomsamplewithoutreplacement implements efficient algorithms for sampling nsamples

integers from a population of size npopulation without replacement

734 Efficient Routines for Sparse Matrices

The sklearnutilssparsefuncs cython module hosts compiled extensions to efficiently process scipy

sparse data

- sparsefuncsmeanvarianceaxis compute the means and variances along a specified axis of a CSR (Compressed Sparse Row)

matrix Used for normalizing the tolerance stopping criterion in sklearnclusterKMeans

- sparsefuncsfastinplacecsrrownormalizel1 andsparsefuncsfast

inplacecsrrownormalizel2 can be used to normalize individual sparse samples to unit L1

or L2 norm as done in sklearnpreprocessingNormalizer

- sparsefuncsinplacecsrcolumnscale can be used to multiply the columns of a CSR (Compressed Sparse Row) matrix

by a constant scale one scale per column Used for scaling features to unit standard deviation in sklearn

preprocessingStandardScaler

735 Graph Routines

- graphssinglesourceshortestpathlength not currently used in scikitlearn Return the

shortest path from a single source to all connected nodes on a graph Code is adapted from networkx

If this is ever needed again it would be far faster to use a single iteration of Dijkstra’s algorithm from

graphshortestpath

- graphshortestpathgraphshortestpath used insklearnmanifoldIsomap Return

the shortest path between all pairs of connected points on a directed or undirected graph Both the Floyd

Warshall algorithm and Dijkstra’s algorithm are available The algorithm is most efficient when the connectivity

matrix is ascipysparsecsrmatrix

736 Testing Functions

- testingassertin testingassertnotin Assertions for container membership Designed for

forward compatibility with Nose 10

- testingassertraisemessage Assertions for checking the error raise message

- testingmockmldataurlopen Mocks the urlopen function to fake requests to mldataorg Used in

tests ofsklearn datasets

- testingalallestimators returns a list of all estimators in scikitlearn to test for consistent behavior

and interfaces

73 Utilities for Developers 2435

scikitlearn user guide Release 0213

737 Multiclass and multilabel utility function

- multiclassismultilabel Helper function to check if the task is a multilabel classification one
- multiclassuniquelabels Helper function to extract an ordered array of unique labels from different formats of target

738 Helper Functions

- genevenslices generator to create npacks of slices going up to n Used in sklearn decompositiondictlearning andsklearnclusterkmeans
- safemask Helper function to convert a mask to the format expected by the numpy array or scipy sparse matrix on which to use it sparse matrices support integer indices only while numpy arrays support both boolean masks and integer indices
- safesqr Helper function for unified squaring 2 of arraylikes matrices and sparse matrices

739 Hash Functions

- murmurhash332 provides a python wrapper for the MurmurHash3x8632 C non cryptographic hash function This hash function is suitable for implementing lookup tables Bloom filters Count Min Sketch feature hashing and implicitly defined sparse random projections
- ```
from sklearnutils import murmurhash332
murmurhash332some feature seed0 384616559
True
murmurhash332some feature seed0 positive True 3910350737
True
```

The sklearnutilsmurmurhash module can also be “cimported” from other cython modules so as to benefit from the high performance of MurmurHash while skipping the overhead of the Python interpreter

7310 Warnings and Exceptions

- deprecated Decorator to mark a function or class as deprecated
- sklearnexceptionsConvergenceWarning Custom warning to catch convergence problems Used

insklearncovariancegraphicallasso

74 How to optimize for speed

The following gives some practical guidelines to help you write efficient code for the scikitlearn project

Note While it is always useful to profile your code so as to check performance assumptions it is also highly recommended to review the literature to ensure that the implemented algorithm is the state of the art for the task before investing into costly implementation optimization

Times and times hours of efforts invested in optimizing complicated implementation details have been rendered irrelevant by the subsequent discovery of simple algorithmic tricks or by using another algorithm altogether that is better suited to the problem

2436 Chapter 7 Developer’s Guide

scikitlearn user guide Release 0213

The section A simple algorithmic trick warm restarts gives an example of such a trick

741 Python Cython or CC

In general the scikitlearn project emphasizes the readability of the source code to make it easy for the project users to dive into the source code so as to understand how the algorithm behaves on their data but also for ease of maintainability by the developers

When implementing a new algorithm is thus recommended to start implementing it in Python using Numpy and Scipy by taking care of avoiding looping code using the vectorized idioms of those libraries In practice this means trying to replace any nested for loops by calls to equivalent Numpy array methods The goal is to avoid the CPU wasting time in the Python interpreter rather than crunching numbers to fit your statistical model It's generally a good idea to consider NumPy and SciPy performance tips <http://scipy.github.io/oldwiki/pages/PerformanceTips>

Sometimes however an algorithm cannot be expressed efficiently in simple vectorized Numpy code In this case the recommended strategy is the following

1 Profile the Python implementation to find the main bottleneck and isolate it in a dedicated module level function This function will be reimplemented as a compiled extension module

2 If there exists a well maintained BSD (Binary Search Decision) or MIT CC implementation of the same algorithm that is not too big you can write a Cython wrapper for it and include a copy of the source code of the library in the scikit learn source tree this strategy is used for the classes svmLinearSVC svmSVC and linearModel

LogisticRegression wrappers for liblinear and libsvm

3 Otherwise write an optimized version of your Python function using Cython directly This strategy is used for the linearModel ElasticNet and linearModelSGDClassifier classes for instance

4 Move the Python version of the function in the tests and use it to check that the results of the compiled extension are consistent with the gold standard easy to debug Python version

5 Once the code is optimized not simple bottleneck spottable by profiling check whether it is possible to have coarse grained parallelism that is amenable to multiprocessing by using the joblibParallel class

When using Cython use either

python setup.py build\_ext in python setup.py install

to generate C files You are responsible for adding ccpp extensions along with build parameters in each submodule setup.py

CC generated files are embedded in distributed stable packages The goal is to make it possible to install scikitlearn stable version on any machine with Python Numpy Scipy and CC compiler

742 Profiling Python code

In order to profile Python code we recommend to write a script that loads and prepare you data and then use the IPython integrated profiler for interactively exploring the relevant part for the code

Suppose we want to profile the Non Negative Matrix Factorization module of scikitlearn Let us setup a new IPython session and load the digits dataset and as in the Recognizing handwritten digits example

In 1 from sklearn.decomposition import NMF (Non-Negative Matrix Factorization)

In 2 from sklearn.datasets import load\_digits

In 3 X = load\_digits.data

74 How to optimize for speed 2437

scikitlearn user guide Release 0213

Before starting the profiling session and engaging in tentative optimization iterations it is important to measure the total execution time of the function we want to optimize without any kind of profiler overhead and save it somewhere for later reference

In 4 timeit NMFncomponents16 tol1e2fitX  
1 loops best of 3 17 s per loop

To have a look at the overall performance profile using the prun magic command

In 5 prun l nmfpy NMFncomponents16 tol1e2fitX  
14496 function calls in1682 CPU seconds  
Ordered by internal time  
List reduced from90 to 9 due to restriction nmfpy  
ncalls tottime percall cumtime percall filename:lineno:funcname  
36 0609 0017 1499 0042 nmfpy151nlssubproblem  
1263 0157 0000 0157 0000 nmfpy18pos  
1 0053 0053 1681 1681 nmfpy352fittransform  
673 0008 0000 0057 0000 nmfpy28norm  
1 0006 0006 0047 0047 nmfpy42initializenmf  
36 0001 0000 0010 0000 nmfpy36sparseness  
30 0001 0000 0001 0000 nmfpy23neg  
1 0000 0000 0000 0000 nmfpy337init  
1 0000 0000 1681 1681 nmfpy461fit

The tottime column is the most interesting it gives to total time spent executing the code of a given function ignoring the time spent in executing the subfunctions The real total time local code subfunction calls is given by the cumtime column

Note the use of the l nmfpy that restricts the output to lines that contains the “nmfpy” string This is useful to have a quick look at the hotspot of the nmf Python module itself ignoring anything else

Here is the beginning of the output of the same command without the l nmfpy filter

In 5 prun NMFncomponents16 tol1e2fitX  
16159 function calls in1840 CPU seconds  
Ordered by internal time  
ncalls tottime percall cumtime percall filename:lineno:funcname  
2833 0653 0000 0653 0000 numpycoredotblasdot  
46 0651 0014 1636 0036 nmfpy151nlssubproblem  
1397 0171 0000 0171 0000 nmfpy18pos  
2780 0167 0000 0167 0000 method sum of numpyndarray  
→objects  
1 0064 0064 1840 1840 nmfpy352fittransform  
1542 0043 0000 0043 0000 method flatten of numpyndarray  
→objects  
337 0019 0000 0019 0000 method all of numpyndarray  
→objects  
2734 0011 0000 0181 0000 fromnumericpy1185sum  
2 0010 0005 0010 0005 numpylinalgpackklitedgesdd  
748 0009 0000 0065 0000 nmfpy28norm

The above results show that the execution is largely dominated by dot products operations delegated to blas Hence there is probably no huge gain to expect by rewriting this code in Cython or CC in this case out of the 17s total execution time almost 07s are spent in compiled code we can consider optimal By rewriting the rest of the Python

2438 Chapter 7 Developer’s Guide

scikitlearn user guide Release 0213

code and assuming we could achieve a 1000 boost on this portion which is highly unlikely given the shallowness of the Python loops we would not gain more than a 24x speedup globally  
Hence major improvements can only be achieved by algorithmic improvements in this particular example eg trying to find operation that are both costly and useless to avoid computing then rather than trying to optimize their implementation

It is however still interesting to check what's happening inside the nlssubproblem function which is the hotspot if we only consider Python code it takes around 100 of the accumulated time of the module In order to better understand the profile of this specific function let us install lineprofiler and wire it to IPython

pip install lineprofiler

•Under IPython 013 first create a configuration profile

ipython profile create

Then register the lineprofiler extension in ipythonprofiledefaultipythonconfigpy

cTerminalIPythonAppextensionsappendlineprofiler

cInteractiveShellAppextensionsappendlineprofiler

This will register the lprun magic command in the IPython terminal application and the other frontends such as qtconsole and notebook

Now restart IPython and let us use this new toy

In 1 from sklearn.datasets import load\_digits

In 2 from sklearn.decomposition.nmf import nlssubproblem NMF (Non-Negative Matrix Factorization)

In 3 X = load\_digits.data

In 4 lprun -f nlssubproblem NMF ncomponents=16 tol=1e-2 fit X

Timer unit 1e06 s

File sklearn.decomposition.nmf.py

Function nlssubproblem at line 137

Total time 173153 s

Line Hits Time Per Hit Time Line Contents

137 def nlssubproblem(V, W, Hinit

↳ tol, maxiter

138 Nonnegative least square

↳ solver

170

171 48 5863 1221 03 if Hinit == 0: any

172 raise ValueError('Negative

↳ values in Hinit passed to NLS solver

173

174 48 139 29 00 H = Hinit

175 48 112141 23363 58 WtV = np.dot(WT, V)

176 48 16144 3363 08 WtW = np.dot(WT, W)

177

178 values justified in the paper

179 48 144 30 00 alpha = 1

180 48 113 24 00 beta = 0.1

74 How to optimize for speed 2439

```
scikitlearn user guide Release 0213
181 638 1880 29 01 forniterinrange1 maxiter
↩→ 1
182 638 195133 3059 102 grad npdotWtW H WtV
183 638 495761 7771 259 projgradient normgradnp
↩→logicalorgrad 0 H 0
184 638 2449 38 01 ifprojgradient tol
185 48 130 27 00 break
186
187 1474 4474 30 02 forinneriter inrange1
↩→20
188 1474 83833 569 44 Hn H alpha grad
189 Hn npwhereHn 0
↩→Hn 0
190 1474 194239 1318 101 Hn posHn
191 1474 48858 331 25 d Hn H
192 1474 150407 1020 78 gradd npsumgrad d
193 1474 515390 3497 269 dQd npsumnpdotWtW
↩→dd
```

By looking at the top values of the Time column it is really easy to pinpoint the most expensive expressions that would deserve additional care

743 Memory usage profiling

You can analyze in detail the memory usage of any Python code with the help of memoryprofiler First install the latest version

```
pip install U memoryprofiler
```

Then setup the magics in a manner similar to lineprofiler

- Under IPython 011 first create a configuration profile

```
ipython profile create
```

Then register the extension in ipythonprofiledefaultipythonconfigpy alongside the line profiler

```
cTerminalIPythonAppextensionsappendmemoryprofiler
```

```
cInteractiveShellAppextensionsappendmemoryprofiler
```

This will register the memit andmprun magic commands in the IPython terminal application and the other frontends such as qtconsole and notebook

mprun is useful to examine linebyline the memory usage of key functions in your program It is very similar to lprun discussed in the previous section For example from the memoryprofilerexamples directory

```
In 1from example import myfunc
```

```
In 2 mprun f myfunc myfunc
```

Filename examplepy

Line Mem usage Increment Line Contents

```
3 profile
```

```
4 597 MB 000 MB defmyfunc
```

scikitlearn user guide Release 0213

5 1361 MB 764 MB a 1 106

6 16620 MB 15259 MB b 2 2107

7 1361 MB 15259 MB delb

8 1361 MB 000 MB returna

Another useful magic that memoryprofiler defines is memit which is analogous to timeit It can be used as follows

In 1 import numpy as np

In 2 memit npzeros1e7

maximum of 3 76402344 MB per loop

For more details see the docstrings of the magics using memit andmprun

744 Performance tips for the Cython developer

If profiling of the Python code reveals that the Python interpreter overhead is larger by one order of magnitude or more than the cost of the actual numerical computation eg for loops over vector components nested evaluation of conditional expression scalar arithmetic it is probably adequate to extract the hotspot portion of the code as a standalone function in a pyx file add static type declarations and then use Cython to generate a C program suitable to be compiled as a Python extension module

The official documentation available at <http://docscython.org> contains a tutorial and reference guide for developing such a module In the following we will just highlight a couple of tricks that we found important in practice on the existing cython codebase in the scikitlearn project

TODO html report type declarations bound checks division by zero checks memory alignment direct blas calls

- <https://www.youtube.com/watch?v=MvkiQgOW8>
- <http://conferencescipy.org/proceedings/SciPy2009/paper1>
- <http://conferencescipy.org/proceedings/SciPy2009/paper2>

Using OpenMP

Since scikitlearn can be built without OpenMP support it's necessary to protect each direct call to OpenMP This can be done using the following syntax

importing OpenMP

IF SKLEARNOPENMPSUPPORTED

cimport openmp

calling OpenMP

IF SKLEARNOPENMPSUPPORTED

maxthreads openmpompgetmaxthreads

ELSE

maxthreads 1

Note Protecting the parallel loop prange is already done by cython

74 How to optimize for speed 2441

scikitlearn user guide Release 0213

745 Profiling compiled extensions

When working with compiled extensions written in CC with a wrapper or directly as Cython extension the default Python profiler is useless we need a dedicated tool to introspect what’s happening inside the compiled extension it self

Using yep and gperftools

Easy profiling without special compilation options use yep

- <https://pypi.org/project/yep/>
- <http://fabianp.net/blog/2011/a-profiler-for-python-extensions>

Using gprof

In order to profile compiled Python extensions one could use gprof after having recompiled the project with gcc pg and using the pythondbg variant of the interpreter on debian ubuntu however this approach requires to also have numpy and scipy recompiled with pg which is rather complicated to get working

Fortunately there exist two alternative profilers that don’t require you to recompile everything

Using valgrind callgrind kcache-grind

kcache-grind

yep can be used to create a profiling report kcache-grind provides a graphical environment to visualize this report

Run yep to profile some python script

python m yep c myfile.py

open myfile.py callgrind with kcache-grind

kcache-grind myfile.py prof

Note yep can be executed with the argument lines or to compile a profiling report ‘line by line’

746 Multicore parallelism using joblib.Parallel

TODO give a simple teaser example here

Checkout the official joblib documentation

- <https://joblib.readthedocs.io>

747 A simple algorithmic trick warm restarts

See the glossary entry for warmstart

2442 Chapter 7 Developer’s Guide



scikitlearn user guide Release 0213

75 Advanced installation instructions

There are different ways to get scikitlearn installed

- Install an official release This is the best approach for most users It will provide a stable version and prebuild packages are available for most platforms
  - Install the version of scikitlearn provided by your operating system or Python distribution This is a quick option for those who have operating systems that distribute scikitlearn It might not provide the latest release version
  - Building the package from source This is best for users who want the latestandgreatest features and aren't afraid of running brandnew code This document describes how to build from source
- Note If you wish to contribute to the project you need to install the latest development version

751 Installing nightly builds

The continuous integration servers of the scikitlearn project build test and upload wheel packages for the most recent Python version on a nightly basis to help users test bleeding edge features or bug fixes

pip install pre f httpssklearnnightlyscdn8secureraxcdncom scikitlearn

752 Building from source

In the vast majority of cases building scikitlearn for development purposes can be done with

pip install cython pytest flake8

Then in the main repository

pip install editable

Please read below for details and more advanced instructions

Dependencies

Scikitlearn requires

- Python 35
- NumPy 111
- SciPy 017
- Joblib 011

Note For installing on PyPy PyPy3v510 Numpy 1140 and scipy 110 are required For PyPy only installa

tion instructions with pip apply

Building Scikitlearn also requires

- Cython 0285

75 Advanced installation instructions 2443

scikitlearn user guide Release 0213

- OpenMP

Note It is possible to build scikitlearn without OpenMP support by setting the SKLEARNNOOPENMP environment variable before cythonization This is not recommended since it will force some estimators to run in sequential mode and their njobs parameter will be ignored

Running tests requires

- pytest 330

Some tests also require pandas

Retrieving the latest code

We use Git for version control and GitHub for hosting our main repository

You can check out the latest sources with the command

`git clone gitgithubcomscikitlearnscikitlearngit`

If you want to build a stable version you can `git checkout VERSION` to get the code for that particular

version or download a zip archive of the version from github

Once you have all the build requirements installed see below for details you can build and install the package in the following way

If you run the development version it is cumbersome to reinstall the package each time you update the sources

Therefore it's recommended that you install in editable mode which allows you to edit the code inplace This builds the extension in place and creates a link to the development directory see the pip docs

`pip install editable`

Note This is fundamentally similar to using the command `python setup.py develop` see the `setuptools`

docs It is however preferred to use pip

Note You will have to rerun

`pip install editable`

every time the source code of a compiled extension is changed for instance when switching branches or pulling changes from upstream Compiled extensions are Cython files ending in `.pyx` or `.pxd`

On Unixlike systems you can equivalently type `make` in from the `toplevel` folder Have a look at the `Makefile` for additional utilities

Mac OSX

The default C compiler `AppleClang` on Mac OSX does not directly support OpenMP The first solution to build scikitlearn is to install another C compiler such as `gcc` or `llvmclang` Another solution is to enable OpenMP support on the default `AppleClang` In the following we present how to configure this second option

You first need to install the OpenMP library

2444 Chapter 7 Developer's Guide

scikitlearn user guide Release 0213

brew install libomp

Then you need to set the following environment variables

export CCusrbinclang

export CXXusrbinclang

export CPPFLAGSCPPFLAGS Xpreprocessor fopenmp

export CFLAGSCFLAGS lusrlocaloptlibompinclude

export CXXFLAGSCXXFLAGS lusrlocaloptlibompinclude

export LDFLAGSLDFLAGS Lusrlocaloptlibomplib lomp

export DYLDLIBRARYPATHHusrlocaloptlibomplib

Finally you can build the package using the standard command

FreeBSD

The clang compiler included in FreeBSD 120 and 112 base systems does not include OpenMP support You need to

install theopenmp library from packages or ports

sudo pkg install openmp

This will install header files in usrlocalinclude and libs inusrlocallib Since these directories are

not searched by default you can set the environment variables to these locations

export CFLAGSCFLAGS lusrlocalinclude

export CXXFLAGSCXXFLAGS lusrlocalinclude

export LDFLAGSLDFLAGS Lusrlocallib lomp

export DYLDLIBRARYPATHHusrlocallib

Finally you can build the package using the standard command

For the upcoming FreeBSD 121 and 113 versions OpenMP will be included in the base system and these steps

will not be necessary

753 Installing build dependencies

Linux

Installing from source without conda requires you to have installed the scikitlearn runtime dependencies Python

development headers and a working CC compiler Under Debianbased operating systems which include Ubuntu

sudo aptget install buillessential python3dev python3setuptools

python3pip

and then

pip3 install numpy scipy cython

Note In order to build the documentation and run the example code contains in this documentation you will need

matplotlib

pip3 install matplotlib

75 Advanced installation instructions 2445

scikitlearn user guide Release 0213

When precompiled wheels are not available for your architecture you can install the system versions

sudo apt-get install cython3 python3numpy python3scipy python3matplotlib

On Red Hat and clones eg CentOS install the dependencies using

sudo yum -y install gcc gcc python-devel numpy scipy

Note To use a high performance BLAS library eg OpenBlas see scipy installation instructions

Windows

To build scikitlearn on Windows you need a working C compiler in addition to numpy scipy and setuptools

The building command depends on the architecture of the Python interpreter 32bit or 64bit You can check the architecture by running the following in cmd or powershell console

python -c import struct; print(struct.calcsize('P') \* 8)

The above commands assume that you have the Python installation folder in your PATH environment variable

You will need Build Tools for Visual Studio 2017

For 64bit Python configure the build environment with

SET DISTUTILS\_USE\_SDK=1

CPROGRAMFILES=x86\_Microsoft Visual

Studio 2017 Build Tools VC Auxiliary Build\vcvarsall.bat x64

And build scikitlearn from this environment

python setup.py install

Replace x64 by x86 to build for 32bit Python

Building binary packages and installers

The wheel package and exe installers can be built with

pip install wheel

python setup.py bdist\_wheel bdist\_wininst bdist\_logos scikitlearn\_logos

The resulting packages are generated in the dist folder

Using an alternative compiler

It is possible to use MinGW a port of GCC to Windows OS as an alternative to MSVC for 32bit Python Not that extensions built with mingw32 can be redistributed as reusable packages as they depend on GCC runtime libraries typically not installed on end users environment

To force the use of a particular compiler pass the compiler flag to the build step

2446 Chapter 7 Developer's Guide

scikitlearn user guide Release 0213

python setup.py build compilermycompiler install  
where mycompiler should be one of mingw32 or msvc  
76 Maintainer coredeveloper information

761 Before a release

1 Update authors table

cd buildtools make authors cd  
and commit

2 Confirm any blockers tagged for the milestone are resolved and that other issues tagged for the milestone can be postponed

3 Ensure the change log and commits correspond within reason and that the change log is reasonably well curated Some tools for these tasks include

- mainttoolssortwhatsnew.py can put what's new entries into sections
- Themainttoolswhatmissingsh script may be used to identify pull requests that were merged but likely missing from What's New

Preparing a bugfix release

Since any commits to a released branch eg 0999X will automatically update the web site documentation it is best to develop a bugfix release with a pull request in which 0999X is the base It also allows you to keep track of any tasks towards release with a TO DO list

Most development of the bug fix release and its documentation should happen in master to avoid asynchrony To select commits from master for use in the bug fix version 09993 you can use

git checkout b release09993 master

git rebase i 0999X

Then pick the commits for release and resolve any issues and create a pull request with 0999X as base Add a commit updating sklearn version Additional commits can be cherry picked into the release0

9993 branch while preparing the release

762 Making a release

1 Update docs

- Edit the docwhatsnewrst file to add release title and commit statistics You can retrieve commit statistics with

git shortlog s 09933 cut f2 sort ignorecase tr n

↵sed s gs

- Update the release date in whatsnewrst
- Edit the docindexrst to change the 'News' entry of the front page

76 Maintainer coredeveloper information 2447

scikitlearn user guide Release 0.21.3

- Note that these changes should be made in master and cherry-picked into the release branch
- 2 On the branch for releasing update the version number in sklearn\_init.py the version variable by removing dev only when ready to release On master increment the version in the same place when branching for release

3 Create the tag and push it

```
git tag a.0.999
git push git@github.com:scikitlearn/scikitlearn.git tags
```

4 Create the source tarball

- Wipe clean your repo
- ```
git clean -x -f
```
- Generate the tarball
- ```
python setup.py sdist
```

The result should be in the dist folder We will upload it later with the wheels Check that you can install it in a new virtualenv and that the tests pass

5 Update the dependency versions and set BUILD\_COMMIT variable to the release tag at

<https://github.com/MacPythons/scikitlearnwheels>

Once the CI has completed successfully collect the generated binary wheel packages and upload them to PyPI by running the following commands in the scikitlearn source folder checked out at the release tag

```
pip install U wheelhouseuploader twine
python setup.py fetchartifacts
```

6 Check the content of the dist folder it should contain all the wheels along with the source tarball "scikit-learn-XXX.tar.gz"

Make sure that you do not have developer versions or older versions of the scikitlearn package in that folder

Upload everything at once to <https://pypi.org>

```
twine upload dist
```

7 For major/minor not bugfix release update the symlink for stable in <https://github.com/scikitlearn/scikitlearn.github.io>

```
cd tmp
git clone --depth 1 --no-checkout git@github.com:scikitlearn/scikitlearn
cd scikitlearn.github.io
echo stable gitinfo: sparse-checkout
git checkout master
ln -sf 0.999 stable
git push origin master
```

The following GitHub checklist might be helpful in a release PR

- update news and what's new date in master and release branch
- create tag
- update dependencies and release tag at <https://github.com/MacPythons/scikitlearnwheels>

scikitlearn user guide Release 0213  
twine the wheels to PyPI when thats green  
httpsgithubcomscikitlearnscikitlearnreleases draft  
confirm bot detected at httpsgithubcomcondaforgecscikitlearnfeedstock  
↩→andwaitformerge  
httpsgithubcomscikitlearnscikitlearnreleases publish  
announce on mailing list  
regenerate Dash docs httpsgithubcomKapeliDashUserContributionstree  
↩→masterdocsetsScikit

763 The scikitlearnorg web site

The scikitlearn web site httpscikitlearnorg is hosted at GitHub but should rarely be updated manually by pushing to the httpsgithubcomscikitlearnscikitlearngithubio repository Most updates can be made by pushing to master for dev or a release branch like 099X from which Circle CI builds and uploads the documentation automatically

764 Travis Cron jobs

From httpsdocstraviscicomusercronjobs Travis CI cron jobs work similarly to the cron utility they run builds at regular scheduled intervals independently of whether any commits were pushed to the repository Cron jobs always fetch the most recent commit on a particular branch and build the project at that state Cron jobs can run daily weekly or monthly which in practice means up to an hour after the selected time span and you cannot set them to run at a specific time

For scikitlearn Cron jobs are used for builds that we do not want to run in each PR As an example the build with the dev versions of numpy and scipy is run as a Cron job Most of the time when this numpydev build fail it is related to a numpy change and not a scikitlearn one so it would not make sense to blame the PR author for the Travis failure The definition of what gets run in the Cron job is done in the travisyaml config file exactly the same way as the other Travis jobs We use a if type cron filter in order for the build to be run only in Cron jobs

The branch targeted by the Cron job and the frequency of the Cron job is set via the web UI at httpswwwtravisciorgscikitlearnscikitlearnsettings

765 Experimental features

Thesklearnexperimental module was introduced in 021 and contains experimental features estimators that are subject to change without deprecation cycle

To create an experimental module you can just copy and modify the content of enablehistgradientboostingpy or enableiterativeimputerpy

Note that the public import path must be to a public subpackage like sklearnensemble orsklearnimpute not just apy module Also the private experimental features that are imported must be in a submodulesubpackage of the public subpackage eg sklearnensemblehistgradientboosting orsklearnimpute iterativepy This is needed so that pickles still work in the future when the features aren’t experimental anymore

Please also write basic tests following those in testenablehistgradientboostingpy

Make sure every userfacing code you write explicitly mentions that the feature is experimental and add a noqa comment to avoid pep8related warnings

To use this experimental feature we need to explicitly ask for it  
from sklearnexperimental import enablehistgradientboosting noqa  
from sklearnensemble import HistGradientBoostingRegressor

76 Maintainer coredeveloper information 2449

scikitlearn user guide Release 0.21.3

For the docs to render properly please also import enablemyexperimentalfeature into conf.py  
else sphinx won't be able to import the corresponding modules Note that using from sklearn.experimental  
import does not work

Note that some experimental classes / functions are not included in the sklearn.experimental module  
sklearn.datasets.fetch\_openml

2450 Chapter 7 Developer's Guide



BIBLIOGRAPHY

M2012 “Machine Learning A Probabilistic Perspective” Murphy K P chapter 1443 pp 492493 The MIT Press 2012

RW2006 Carl Eduard Rasmussen and Christopher KI Williams “Gaussian Processes for Machine Learning” MIT Press 2006 Link to an official complete PDF version of the book here

B1999 L Breiman “Pasting small votes for classification in large databases and online” Machine Learning 361 85103 1999

B1996 L Breiman “Bagging predictors” Machine Learning 242 123140 1996

H1998 T Ho “The random subspace method for constructing decision forests” Pattern Analysis and Machine Intelligence 208 832844 1998

LG2012 G Louppe and P Geurts “Ensembles on Random Patches” Machine Learning and Knowledge Discovery in Databases 346361 2012

B2001 12 Breiman “Random Forests” Machine Learning 451 532 2001

B1998 12 Breiman “Arcing Classifiers” Annals of Statistics 1998

L2014 G Louppe “Understanding Random Forests From Theory to Practice” PhD Thesis U of Liege 2014

FS1995 Y Freund and R Schapire “A DecisionTheoretic Generalization of OnLine Learning and an Application to Boosting” 1997

ZZRH2009 J Zhu H Zou S Rosset T Hastie “Multiclass AdaBoost” 2009

D1997 8 Drucker “Improving Regressors using Boosting Techniques” 1997

HTF T Hastie R Tibshirani and J Friedman “Elements of Statistical Learning Ed 2” Springer 2009

VEB2009 Vinh Epps and Bailey 2009 “Information theoretic measures for clusterings comparison” Proceedings of the 26th Annual International Conference on Machine Learning ICML ‘09 doi10114515533741553511 ISBN 9781605585161

VEB2010 Vinh Epps and Bailey 2010 “Information Theoretic Measures for Clusterings Comparison Variants Properties Normalization and Correction for Chance” JMLR httpjmlr.csail.mit.edu/papers/volume11/vinh10a/vinh10a.pdf

YAT2016 Yang Algesheimer and Tessone 2016 “A comparative analysis of community detection algorithms on artificial networks” Scientific Reports 6 30750 doi101038/srep30750

B2011 Identification and Characterization of Events in Social Media Hila Becker PhD Thesis

Mrl09 “Online Dictionary Learning for Sparse Coding” J Mairal F Bach J Ponce G Sapiro 2009

Jen09 “Structured Sparse Principal Component Analysis” R Jenatton G Obozinski F Bach 2009 2451

scikitlearn user guide Release 0213

R45f14345c0001 12 Breiman “Random Forests” Machine Learning 451 532 2001

Rf91cab2dc4271 12 Breiman “Random Forests” Machine Learning 451 532 2001

Rf91cab2dc4272 P Geurts D Ernst and L Wehenkel “Extremely randomized trees” Machine Learning 631 342 2006

Rc8f28bfad63f1 P Geurts D Ernst and L Wehenkel “Extremely randomized trees” Machine Learning 631 342 2006

Ra7d0c8995fbc1 P Geurts D Ernst and L Wehenkel “Extremely randomized trees” Machine Learning 631 342 2006

Guyon2015 I Guyon K Bennett G Cawley HJ Escalante S Escalera TK Ho N Macià B Ray M Saeed

AR Statnikov E Viegas Design of the 2015 ChaLearn AutoML Challenge IJCNN 2015

Mosley2013 L Mosley A balanced approach to the multiclass imbalance problem IJCV 2010

Kelleher2015 John D Kelleher Brian Mac Namee Aoife D’Arcy Fundamentals of Machine Learning for Predictive Data Analytics Algorithms Worked Examples and Case Studies 2015

Urbanowicz2015 Urbanowicz RJ Moore JH ExSTraCS 20 description and evaluation of a scalable learning classifier system Evol Intel 2015 8 89

Manning2008 CD Manning P Raghavan H Schütze Introduction to Information Retrieval 2008

Everingham2010 M Everingham L Van Gool CKI Williams J Winn A Zisserman The Pascal Visual Object Classes VOC Challenge IJCV 2010

Davis2006 J Davis M Goadrich The Relationship Between PrecisionRecall and ROC Curves ICML 2006

Flach2015 PA Flach M Kull PrecisionRecallGain Curves PR Analysis Done Right NIPS 2015

HTF2009 T Hastie R Tibshirani and J Friedman The Elements of Statistical Learning Second Edition Section 10132 Springer 2009

Mol2019 C Molnar Interpretable Machine Learning Section 51 2019

NQY18 J Nothman H Qin and R Yurchak 2018 “Stop Word Lists in Free Opensource Software Packages” In Proc Workshop for NLP Open Source Software

RR2007 “Random features for largescale kernel machines” Rahimi A and Recht B Advances in neural information processing 2007

LS2010 “Random Fourier approximations for skewed multiplicative histogram kernels” Random Fourier approximations for skewed multiplicative histogram kernels Lecture Notes for Computer Sciencd DAGM

VZ2010 “Efficient additive kernels via explicit feature maps” Vedaldi A and Zisserman A Computer Vision and Pattern Recognition 2010

VVZ2010 “Generalized RBF (Radial Basis Function) feature maps for Efficient Detection” Vempati S and Vedaldi A and Zisserman A and Jawahar CV (Cross-Validation) 2010

R57cf438d70601 Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers B Zadrozny C Elkan ICML 2001

R57cf438d70602 Transforming Classifier Scores into Accurate Multiclass Probability Estimates B Zadrozny C Elkan KDD 2002

R57cf438d70603 Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods J Platt 1999

R57cf438d70604 Predicting Good Probabilities with Supervised Learning A NiculescuMizil R Caruana ICML 2005

2452 Bibliography

scikitlearn user guide Release 0213

R2c55e37003fe1 Ankerst Mihael Markus M Breunig HansPeter Kriegel and Jörg Sander “OPTICS ordering points to identify the clustering structure” ACM SIGMOD Record 28 no 2 1999 4960

R2c55e37003fe2 Schubert Erich Michael Gertz “Improving the Cluster Structure Extracted from OPTICS Plots” Proc of the Conference “Lernen Wissen Daten Analysen” LWDA 2018 318329

1 Ankerst Mihael Markus M Breunig HansPeter Kriegel and Jörg Sander “OPTICS ordering points to identify the clustering structure” ACM SIGMOD Record 28 no 2 1999 4960

R68ae096da0e41 Rousseeuw PJ Van Driessen K “A fast algorithm for the minimum covariance determinant estimator” Technometrics 413 212 1999

RVD A Fast Algorithm for the Minimum Covariance Determinant Estimator 1999 American Statistical Association and the American Society for Quality TECHNOMETRICS

RVDriessen A Fast Algorithm for the Minimum Covariance Determinant Estimator 1999 American Statistical Association and the American Society for Quality TECHNOMETRICS

R9f63e655f7bdRousseeuw1984 P J Rousseeuw Least median of squares regression J Am Stat Ass 79871 1984

R9f63e655f7bdRousseeuw A Fast Algorithm for the Minimum Covariance Determinant Estimator 1999 American Statistical Association and the American Society for Quality TECHNOMETRICS

R9f63e655f7bdButlerDavies R W Butler P L Davies and M Jhun Asymptotics For The Minimum Covariance Determinant Estimator The Annals of Statistics 1993 V ol 21 No 3 13851400

RVD A Fast Algorithm for the Minimum Covariance Determinant Estimator 1999 American Statistical Association and the American Society for Quality TECHNOMETRICS

RVDriessen A Fast Algorithm for the Minimum Covariance Determinant Estimator 1999 American Statistical Association and the American Society for Quality TECHNOMETRICS

1 Dhillon I S 2001 August Coclustering documents and words using bipartite spectral graph partitioning In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining pp 269274 ACM

1 Kluger Y Basri R Chang J T Gerstein M 2003 Spectral biclustering of microarray data coclustering genes and conditions Genome research 134 703716

1 I Guyon “Design of experiments for the NIPS 2003 variable selection benchmark” 2003

1 J Friedman “Multivariate adaptive regression splines” The Annals of Statistics 19 1 pages 167 1991

2 L Breiman “Bagging predictors” Machine Learning 24 pages 123140 1996

1 J Friedman “Multivariate adaptive regression splines” The Annals of Statistics 19 1 pages 167 1991

2 L Breiman “Bagging predictors” Machine Learning 24 pages 123140 1996

1 J Friedman “Multivariate adaptive regression splines” The Annals of Statistics 19 1 pages 167 1991

2 L Breiman “Bagging predictors” Machine Learning 24 pages 123140 1996

1 10 Zhu H Zou S Rosset T Hastie “Multiclass AdaBoost” 2009

1 T Hastie R Tibshirani and J Friedman “Elements of Statistical Learning Ed 2” Springer 2009

1 G Celeux M El Anbari JM Marin C P Robert “Regularization in regression comparing Bayesian and frequentist methods in a poorly informative situation” 2009

1 S Marsland “Machine Learning An Algorithmic Perspective” Chapter 10 2009 <http://seatmassey.ac.nz/personalsrmarsland/Code10/lepy>

R33e4ec8c4ad51 Y Freund R Schapire “A DecisionTheoretic Generalization of onLine Learning and an Application to Boosting” 1995

Bibliography 2453

scikitlearn user guide Release 0213

R33e4ec8c4ad52 10 Zhu H Zou S Rosset T Hastie “Multiclass AdaBoost” 2009

R0c261b7dee9d1 Y Freund R Schapire “A DecisionTheoretic Generalization of onLine Learning and an Ap  
plication to Boosting” 1995

R0c261b7dee9d2 8 Drucker “Improving Regressors using Boosting Techniques” 1997

Rb1846455d0e51 L Breiman “Pasting small votes for classification in large databases and online” Machine  
Learning 361 85103 1999

Rb1846455d0e52 L Breiman “Bagging predictors” Machine Learning 242 123140 1996

Rb1846455d0e53 T Ho “The random subspace method for constructing decision forests” Pattern Analysis and  
Machine Intelligence 208 832844 1998

Rb1846455d0e54 G Louppe and P Geurts “Ensembles on Random Patches” Machine Learning and Knowledge  
Discovery in Databases 346361 2012

R4d113ba76fc01 L Breiman “Pasting small votes for classification in large databases and online” Machine  
Learning 361 85103 1999

R4d113ba76fc02 L Breiman “Bagging predictors” Machine Learning 242 123140 1996

R4d113ba76fc03 T Ho “The random subspace method for constructing decision forests” Pattern Analysis and  
Machine Intelligence 208 832844 1998

R4d113ba76fc04 G Louppe and P Geurts “Ensembles on Random Patches” Machine Learning and Knowledge  
Discovery in Databases 346361 2012

Rd7ae0a2ae6881 Liu Fei Tony Ting Kai Ming and Zhou ZhiHua “Isolation forest” Data Mining 2008  
ICDM’08 Eighth IEEE International Conference on

Rd7ae0a2ae6882 Liu Fei Tony Ting Kai Ming and Zhou ZhiHua “Isolationbased anomaly detection” ACM  
Transactions on Knowledge Discovery from Data TKDD 61 2012 3

R6e47e53bacbd1 P Geurts D Ernst and L Wehenkel “Extremely randomized trees” Machine Learning 631  
342 2006

R6e47e53bacbd2 Moosmann F and Triggs B and Jurie F “Fast discriminative visual codebooks using random  
ized clustering forests” NIPS 2007

R1b90ac3ca370Yates2011 R BaezaYates and B RibeiroNeto 2011 Modern Information Retrieval Addison  
Wesley pp 6874

R1b90ac3ca370MRS2008 CD Manning P Raghavan and H Schütze 2008 Introduction to Information Re  
trieval Cambridge University Press pp 118120

Re310f679c81e1 Guyon I Weston J Barnhill S Vapnik V “Gene selection for cancer classification using  
support vector machines” Mach Learn 4613 389–422 2002

R6f4d61ceb4111 Guyon I Weston J Barnhill S Vapnik V “Gene selection for cancer classification using  
support vector machines” Mach Learn 4613 389–422 2002

1 Mutual Information on Wikipedia

2 A Kraskov H Stogbauer and P Grassberger “Estimating mutual information” Phys Rev E 69 2004

3 B C Ross “Mutual Information between Discrete and Continuous Data Sets” PLoS ONE 92 2014

4 L F Kozachenko N N Leonenko “Sample Estimate of the Entropy of a Random Vector Probl Peredachi Inf  
232 1987 916

1 Mutual Information on Wikipedia

2 A Kraskov H Stogbauer and P Grassberger “Estimating mutual information” Phys Rev E 69 2004

2454 Bibliography

scikitlearn user guide Release 0213

3 B C Ross “Mutual Information between Discrete and Continuous Data Sets” PLoS ONE 92 2014

4 L F Kozachenko N N Leonenko “Sample Estimate of the Entropy of a Random Vector” Probl Peredachi Inf 232 1987 916

Rcd31b817a31e1 Stef van Buuren Karin GroothuisOudshoorn 2011 “mice Multivariate Imputation by Chained Equations in R” Journal of Statistical Software 45 167

Rcd31b817a31e2 S F Buck 1960 “A Method of Estimation of Missing Values in Multivariate Data Suitable for use with an Electronic Computer” Journal of the Royal Statistical Society 222 302306

Re4616ef910fb1 Peter J Huber Elvezio M Ronchetti Robust Statistics Concomitant scale estimates pg 172

Re4616ef910fb2 Art B Owen 2006 A robust hybrid of lasso and ridge regression <https://statweb.stanford.edu/owen/reportshhupdf>

R80ce5b25cf9d1 <https://en.wikipedia.org/wiki/RANSAC>

R80ce5b25cf9d2 <https://www.sricomssites/default/files/publications/ransac/publication.pdf>

R80ce5b25cf9d3 <http://www.bmva.org/bmvc/2009/Papers/Paper355/Paper355.pdf>

1 “Least Angle Regression” Efron et al <http://statweb.stanford.edu/tibs/ftp/lars.pdf>

2 Wikipedia entry on the Leastangle regression

3 Wikipedia entry on the Lasso

1 “Least Angle Regression” Efron et al <http://statweb.stanford.edu/tibs/ftp/lars.pdf>

2 Wikipedia entry on the Leastangle regression

3 Wikipedia entry on the Lasso

R7f4d308f50541 Tenenbaum JB De Silva V Langford JC A global geometric framework for nonlinear dimensionality reduction Science 290 5500

R62e36dd1b0561 Roweis S Saul L Nonlinear dimensionality reduction by locally linear embedding Science 2902323 2000

R62e36dd1b0562 Donoho D Grimes C Hessian eigenmaps Locally linear embedding techniques for high dimensional data Proc Natl Acad Sci U S A 1005591 2003

R62e36dd1b0563 Zhang Z Wang J MLLE Modified Locally Linear Embedding Using Multiple Weights <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.703.82>

R62e36dd1b0564 Zhang Z Zha H Principal manifolds and nonlinear dimensionality reduction via tangent space alignment Journal of Shanghai Univ 8406 2004

1 Roweis S Saul L Nonlinear dimensionality reduction by locally linear embedding Science 2902323 2000

2 Donoho D Grimes C Hessian eigenmaps Locally linear embedding techniques for highdimensional data Proc Natl Acad Sci U S A 1005591 2003

3 Zhang Z Wang J MLLE Modified Locally Linear Embedding Using Multiple Weights <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.703.82>

4 Zhang Z Zha H Principal manifolds and nonlinear dimensionality reduction via tangent space alignment Journal of Shanghai Univ 8406 2004

1 Wikipedia entry for the Average precision

1 Brodersen KH Ong CS Stephan KE Buhmann JM 2010 The balanced accuracy and its posterior distribution Proceedings of the 20th International Conference on Pattern Recognition 312124

2 John D Kelleher Brian Mac Namee Aoife D’Arcy 2015 Fundamentals of Machine Learning for Predictive Data Analytics Algorithms Worked Examples and Case Studies Bibliography 2455

scikitlearn user guide Release 0213

1 Wikipedia entry for the Brier score

1 J Cohen 1960 "A coefficient of agreement for nominal scales" Educational and Psychological Measurement 2013746 doi101177001316446002000104

2 R Artstein and M Poesio 2008 "Intercoder agreement for computational linguistics" Computational Linguistics 344555596

3 Wikipedia entry for the Cohen's kappa

1 Wikipedia entry for the Confusion matrix Wikipedia and other references may use a different convention for axes

1 Wikipedia entry for the F1score

1 R BaezaYates and B RibeiroNeto 2011 Modern Information Retrieval Addison Wesley pp 327328

2 Wikipedia entry for the F1score

1 Grigorios Tsoumakas Ioannis Katakis MultiLabel Classification An Overview International Journal of Data Warehousing Mining 33 113 JulySeptember 2007

2 Wikipedia entry on the Hamming distance

1 Wikipedia entry on the Hinge loss

2 Koby Crammer Yoram Singer On the Algorithmic Implementation of Multiclass Kernelbased Vector Machines Journal of Machine Learning Research 2 2001 265292

3 L1 AND L2 Regularization for Multiclass Hinge Loss Models by Robert C Moore John DeNero

1 Wikipedia entry for the Jaccard index

1 Baldi Brunak Chauvin Andersen and Nielsen 2000 Assessing the accuracy of prediction algorithms for classification an overview

2 Wikipedia entry for the Matthews Correlation Coefficient

3 Gorodkin 2004 Comparing two Kcategory assignments by a Kcategory correlation coefficient

4 Jurman Riccadonna Furlanello 2012 A Comparison of MCC and CEN Error Measures in MultiClass Prediction

1 Wikipedia entry for the Precision and recall

2 Wikipedia entry for the F1score

3 Discriminative Methods for Multilabeled Classification Advances in Knowledge Discovery and Data Mining 2004 pp 2230 by Shantanu Godbole Sunita Sarawagi

1 Wikipedia entry for the Receiver operating characteristic

2 Fawcett T An introduction to ROC analysisj Pattern Recognition Letters 2006 278861874

3 Analyzing a portion of the ROC curve McClish 1989

1 Wikipedia entry for the Receiver operating characteristic

2 Fawcett T An introduction to ROC analysisj Pattern Recognition Letters 2006 278861874

1 Wikipedia entry on the Coefficient of determination

1 Tsoumakas G Katakis I Vlahavas I 2010 Mining multilabel data In Data mining and knowledge discovery handbook pp 667685 Springer US

1 Tsoumakas G Katakis I Vlahavas I 2010 Mining multilabel data In Data mining and knowledge discovery handbook pp 667685 Springer US

2456 Bibliography

scikitlearn user guide Release 0213

1 Vinh Epps and Bailey 2010 Information Theoretic Measures for Clusterings Comparison Variants Properties Normalization and Correction for Chance JMLR

2 Wikipedia entry for the Adjusted Mutual Information

Hubert1985 L Hubert and P Arabie Comparing Partitions Journal of Classification 1985 <https://links.springer.com/article/10.1007/2F01908075>

wk <https://en.wikipedia.org/wiki/RandindexAdjustedRandindex>

1 T Calinski and J Harabasz 1974 "A dendrite method for cluster analysis" Communications in Statistics

1 Davies David L Bouldin Donald W 1979 "A Cluster Separation Measure" IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI1 2 224227

1 Andrew Rosenberg and Julia Hirschberg 2007 VMeasure A conditional entropybased external cluster evaluation measure

1 E B Fowlkes and C L Mallows 1983 "A method for comparing two hierarchical clusterings" Journal of the American Statistical Association

2 Wikipedia entry for the FowlkesMallows Index

1 Andrew Rosenberg and Julia Hirschberg 2007 VMeasure A conditional entropybased external cluster evaluation measure

1 Peter J Rousseeuw 1987 "Silhouettes a Graphical Aid to the Interpretation and Validation of Cluster Analysis" Computational and Applied Mathematics 20 5365

2 Wikipedia entry on the Silhouette Coefficient

1 Peter J Rousseeuw 1987 "Silhouettes a Graphical Aid to the Interpretation and Validation of Cluster Analysis" Computational and Applied Mathematics 20 5365

2 Wikipedia entry on the Silhouette Coefficient

1 Andrew Rosenberg and Julia Hirschberg 2007 VMeasure A conditional entropybased external cluster evaluation measure

R16529824bff21 Bishop Christopher M 2006 "Pattern recognition and machine learning" V ol 4 No 4 New York Springer

R16529824bff22 Hagai Attias 2000 "A Variational Bayesian Framework for Graphical Models" In Advances in Neural Information Processing Systems 12

R16529824bff23 Blei David M and Michael I Jordan 2006 "Variational inference for Dirichlet process mixtures" Bayesian analysis 11

R2eddaaec08491 "Solving multiclass learning problems via errorcorrecting output codes" Dietterich T Bakiri G Journal of Artificial Intelligence Research 2 1995

R2eddaaec08492 "The error coding method and PICTs" James G Hastie T Journal of Computational and Graphical statistics 7 1998

R2eddaaec08493 "The Elements of Statistical Learning" Hastie T Tibshirani R Friedman J page 606 second edition 2008

Rca479bb498411 Breunig M M Kriegel H P Ng R T Sander J 2000 May LOF identifying density based local outliers In ACM sigmod record

Rf9b6baee82291 J Goldberger G Hinton S Roweis R Salakhutdinov "Neighbourhood Components Analysis" Advances in Neural Information Processing Systems 17 513520 2005 <http://www.cs.nyu.edu/roweis/papers/ncanips.pdf>

Bibliography 2457

scikitlearn user guide Release 0213

Rf9b6baee82292 Wikipedia entry on Neighborhood Components Analysis [https://en.wikipedia.org/wiki/Neighbourhood\\_components\\_analysis](https://en.wikipedia.org/wiki/Neighbourhood_components_analysis)

Rf3e1504535de1 IK Yeo and RA Johnson “A new family of power transformations to improve normality or symmetry” *Biometrika* 874 pp954959 2000

Rf3e1504535de2 GEP Box and DR Cox “An Analysis of Transformations” *Journal of the Royal Statistical Society B* 26 211252 1964

1 IK Yeo and RA Johnson “A new family of power transformations to improve normality or symmetry” *Biometrika* 874 pp954959 2000

2 GEP Box and DR Cox “An Analysis of Transformations” *Journal of the Royal Statistical Society B* 26 211252 1964

R0fecf191e4b81 Ping Li T Hastie and K W Church 2006 “Very Sparse Random Projections” <https://web.stanford.edu/hastie/Papers/PingKDD06rppdf>

R0fecf191e4b82 D Achlioptas 2001 “Databasefriendly random projections” <https://users.soe.ucsc.edu/optas/papers/jlpdf>

1 [https://en.wikipedia.org/wiki/Johnson%E28093Lindenstrauss\\_lemma](https://en.wikipedia.org/wiki/Johnson%E28093Lindenstrauss_lemma)

2 Sanjoy Dasgupta and Anupam Gupta 1999 “An elementary proof of the JohnsonLindenstrauss Lemma” <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.453.654>

R20c70293ef721 LIBSVM A Library for Support Vector Machines

R20c70293ef722 Platt John 1999 “Probabilistic outputs for support vector machines and comparison to regularized likelihood methods”

Rb1ec977cd3071 [https://en.wikipedia.org/wiki/Decision\\_tree\\_learning](https://en.wikipedia.org/wiki/Decision_tree_learning)

Rb1ec977cd3072 L Breiman J Friedman R Olshen and C Stone “Classification and Regression Trees” Wadsworth Belmont CA 1984

Rb1ec977cd3073 T Hastie R Tibshirani and J Friedman “Elements of Statistical Learning” Springer 2009

Rb1ec977cd3074 L Breiman and A Cutler “Random Forests” <https://www.stat.berkeley.edu/~breiman/RandomForests/scchome.htm>

Ra37b7e3adb191 [https://en.wikipedia.org/wiki/Decision\\_tree\\_learning](https://en.wikipedia.org/wiki/Decision_tree_learning)

Ra37b7e3adb192 L Breiman J Friedman R Olshen and C Stone “Classification and Regression Trees” Wadsworth Belmont CA 1984

Ra37b7e3adb193 T Hastie R Tibshirani and J Friedman “Elements of Statistical Learning” Springer 2009

Ra37b7e3adb194 L Breiman and A Cutler “Random Forests” <https://www.stat.berkeley.edu/~breiman/RandomForests/scchome.htm>

Rdd99a0224c6e1 P Geurts D Ernst and L Wehenkel “Extremely randomized trees” *Machine Learning* 631 342 2006

R4939d63d5a491 P Geurts D Ernst and L Wehenkel “Extremely randomized trees” *Machine Learning* 631 342 2006

1 Wikipedia entry for the Jaccard index

2458 Bibliography























































































