

2nd Meet

How was your week?

□ 오늘 공부할 것은

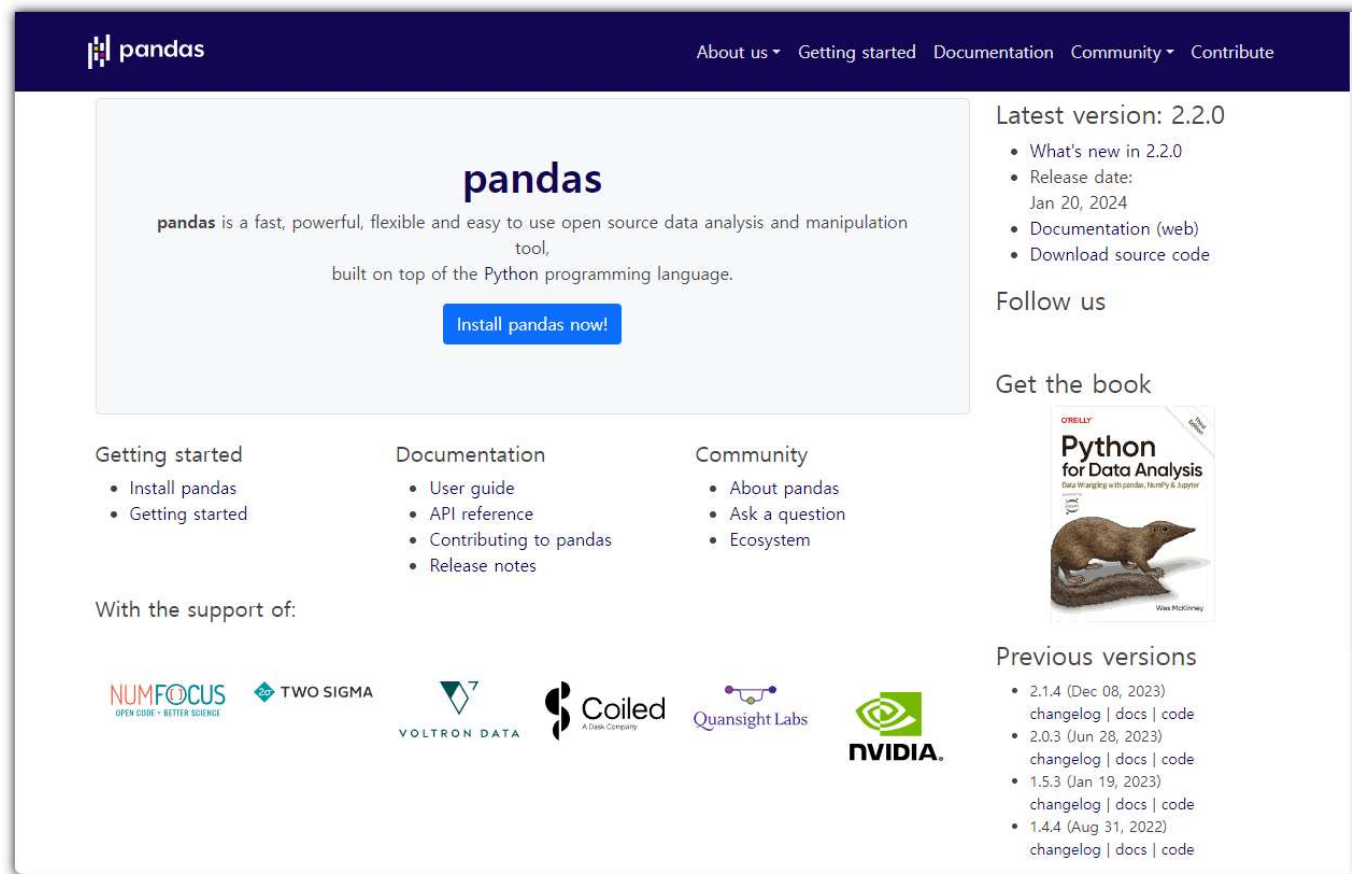
04. 데이터 핸들링 - 판다스	39
판다스 시작 - 파일을 DataFrame으로 로딩, 기본 API	40
DataFrame과 리스트, 딕셔너리, 넘파이 ndarray 상호 변환	49
DataFrame의 칼럼 데이터 세트 생성과 수정	53
DataFrame 데이터 삭제	55
Index 객체	58
데이터 선택 및 필터링	62
정렬, Aggregation 함수, GroupBy 적용	73
결손 데이터 처리하기	77
apply lambda 식으로 데이터 가공	80

01. 사이킷런 소개와 특징	85
02. 첫 번째 머신러닝 만들어 보기 - 붓꽃 품종 예측하기	86
03. 사이킷런의 기반 프레임워크 익히기	91
Estimator 이해 및 fit(), predict() 메서드	91
사이킷런의 주요 모듈	92
내장된 예제 데이터 세트	94
04. Model Selection 모듈 소개	98
학습/테스트 데이터 세트 분리 - train_test_split()	98
교차 검증	100
GridSearchCV - 교차 검증과 최적 하이퍼 파라미터 튜닝을 한 번에	111

Now, It's your turn ! ! !

□ pandas

- <https://pandas.pydata.org/>
- <https://github.com/whatwant-school/python-ml/blob/main/02-pandas.ipynb>



The screenshot shows the pandas website homepage. At the top is a dark blue navigation bar with the pandas logo and links for 'About us', 'Getting started', 'Documentation', 'Community', and 'Contribute'. The main content area has a light blue background with the pandas logo and a description: 'pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.' Below this is a blue button that says 'Install pandas now!'. To the right, there's a section for the 'Latest version: 2.2.0' with a list of links: 'What's new in 2.2.0', 'Release date: Jan 20, 2024', 'Documentation (web)', and 'Download source code'. Below that is a 'Follow us' section. Further down is a 'Get the book' section featuring the cover of the book 'Python for Data Analysis' by Wes McKinney. At the bottom, there are three columns of links: 'Getting started' (Install pandas, Getting started), 'Documentation' (User guide, API reference, Contributing to pandas, Release notes), and 'Community' (About pandas, Ask a question, Ecosystem). Below these columns is a 'With the support of:' section featuring logos for NUMFOCUS, TWO SIGMA, VOLTRON DATA, Coiled, Quansight Labs, and NVIDIA. On the right side, there is a 'Previous versions' section with a list of versions and links to changelogs, docs, and code.

pandas

pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool,
built on top of the Python programming language.

[Install pandas now!](#)

Latest version: 2.2.0

- What's new in 2.2.0
- Release date: Jan 20, 2024
- Documentation (web)
- Download source code

Follow us

Get the book

Python for Data Analysis
Data Wrangling with pandas, NumPy & Jupyter
Wes McKinney

Getting started

- Install pandas
- Getting started

Documentation

- User guide
- API reference
- Contributing to pandas
- Release notes

Community

- About pandas
- Ask a question
- Ecosystem

With the support of:


NUMFOCUS OPEN CODE • BETTER SCIENCE TWO SIGMA VOLTRON DATA Coiled A Data Company Quansight Labs NVIDIA

Previous versions

- 2.1.4 (Dec 08, 2023)
[changelog](#) | [docs](#) | [code](#)
- 2.0.3 (Jun 28, 2023)
[changelog](#) | [docs](#) | [code](#)
- 1.5.3 (Jan 19, 2023)
[changelog](#) | [docs](#) | [code](#)
- 1.4.4 (Aug 31, 2022)
[changelog](#) | [docs](#) | [code](#)

❑ scikit-learn

- <https://scikit-learn.org/stable/>

Install User Guide API Examples Community More ▾Go

scikit-learn

Machine Learning in Python

[Getting Started](#)[Release Highlights for 1.4](#)[GitHub](#)

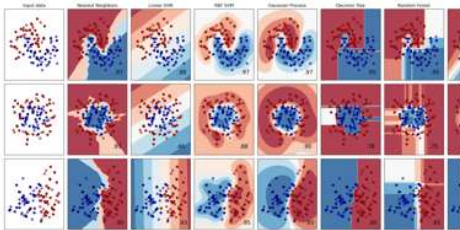
- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.

Algorithms: Gradient boosting, nearest neighbors, random forest, logistic regression, and more...



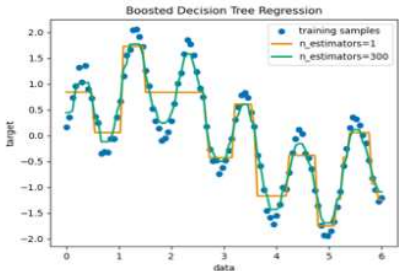
Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: Gradient boosting, nearest neighbors, random forest, ridge, and more...




Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, HDBSCAN, hierarchical clustering, and more...



Examples

[Prev](#) [Up](#) [Next](#)
scikit-learn 1.4.0
[Other versions](#)

Please [cite us](#) if you use the software.

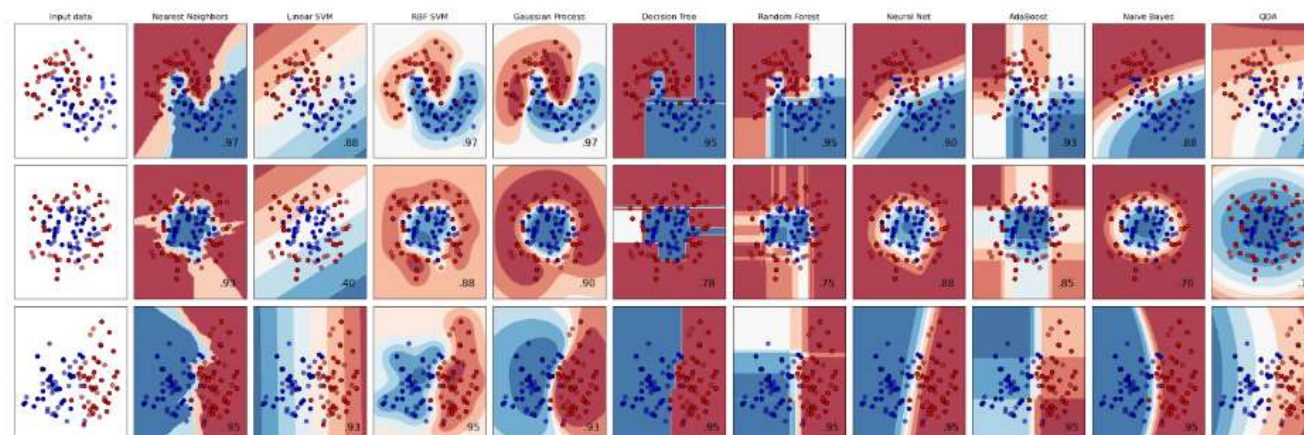
Classifier comparison

Classifier comparison

A comparison of several classifiers in scikit-learn 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 high-dimensional 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.

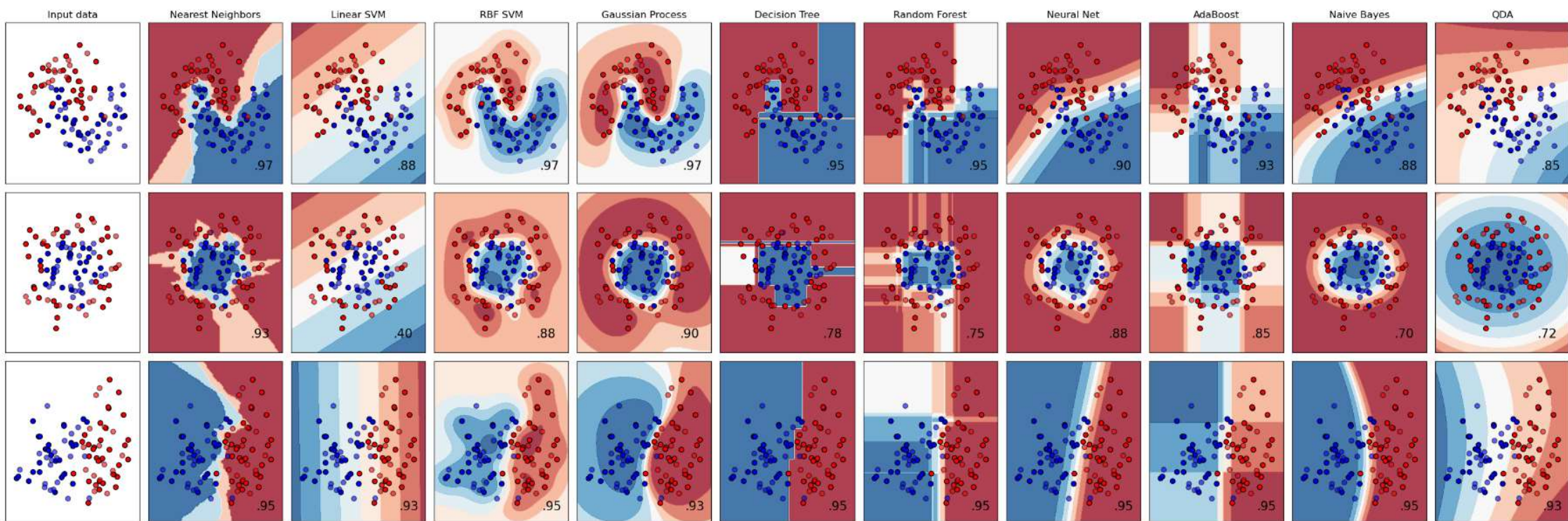
The plots show training points in solid colors and testing points semi-transparent. The lower right shows the classification accuracy on the test set.



```
# Code source: Gaël Varoquaux
#               Andreas Müller
# Modified for documentation by Jaques Grobler
# License: BSD 3 clause
```

```
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.colors import ListedColormap
```

[Toggle Menu](#)



[Prev](#) [Up](#) [Next](#)**scikit-learn 1.4.0**[Other versions](#)

Please **cite us** if you use the software.

User Guide**1. Supervised learning**

- 1.1. Linear Models
- 1.2. Linear and Quadratic Discriminant Analysis
- 1.3. Kernel ridge regression
- 1.4. Support Vector Machines
- 1.5. Stochastic Gradient Descent
- 1.6. Nearest Neighbors
- 1.7. Gaussian Processes
- 1.8. Cross decomposition
- 1.9. Naive Bayes
- 1.10. Decision Trees
- 1.11. Ensembles: Gradient boosting, random forests, bagging, voting, stacking
- 1.12. Multiclass and multioutput algorithms
- 1.13. Feature selection
- 1.14. Semi-supervised learning
- 1.15. Isotonic regression
- 1.16. Probability calibration
- 1.17. Neural network models (supervised)
- 2. Unsupervised learning
- 3. Model selection and evaluation

1. Supervised learning

1.1. Linear Models

- 1.1.1. Ordinary Least Squares
- 1.1.2. Ridge regression and classification
- 1.1.3. Lasso
- 1.1.4. Multi-task Lasso
- 1.1.5. Elastic-Net
- 1.1.6. Multi-task Elastic-Net
- 1.1.7. Least Angle Regression
- 1.1.8. LARS Lasso
- 1.1.9. Orthogonal Matching Pursuit (OMP)
- 1.1.10. Bayesian Regression
- 1.1.11. Logistic regression
- 1.1.12. Generalized Linear Models
- 1.1.13. Stochastic Gradient Descent - SGD
- 1.1.14. Perceptron
- 1.1.15. Passive Aggressive Algorithms
- 1.1.16. Robustness regression: outliers and modeling errors
- 1.1.17. Quantile Regression
- 1.1.18. Polynomial regression: extending linear models with basis functions

1.2. Linear and Quadratic Discriminant Analysis

- 1.2.1. Dimensionality reduction using Linear Discriminant Analysis
- 1.2.2. Mathematical formulation of the LDA and QDA classifiers
- 1.2.3. Mathematical formulation of LDA dimensionality reduction
- 1.2.4. Shrinkage and Covariance Estimator

1.10. Decision Trees

1.10.1. Classification

1.10.2. Regression

1.10.3. Multi-output problems

1.10.4. Complexity

1.10.5. Tips on practical use

1.10.6. Tree algorithms: ID3, C4.5, C5.0 and CART

1.10.7. Mathematical formulation

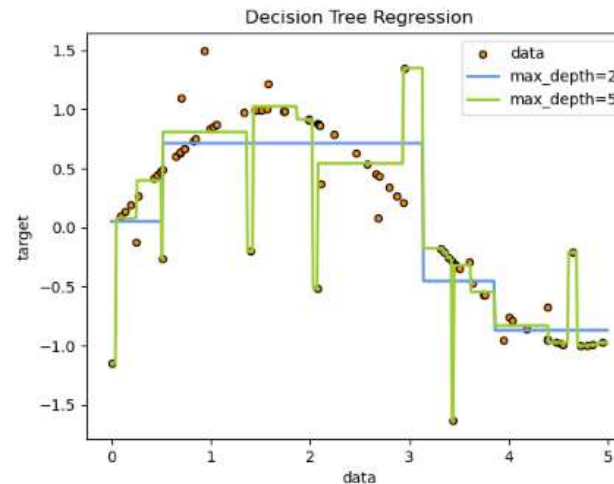
1.10.8. Missing Values Support

1.10.9. Minimal Cost-Complexity Pruning

1.10. Decision Trees

Decision Trees (DTs) are a non-parametric supervised learning method used for [classification](#) and [regression](#). 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. A tree can be seen as a piecewise constant approximation.

For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else 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 visualized.
- Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Some tree and algorithm combinations support [missing values](#).

1.10. Decision Trees

1.10.1. Classification

1.10.2. Regression

1.10.3. Multi-output problems

1.10.4. Complexity

1.10.5. Tips on practical use

1.10.6. Tree algorithms: ID3, C4.5, C5.0 and CART

1.10.7. Mathematical formulation

1.10.8. Missing Values Support

1.10.9. Minimal Cost-Complexity Pruning

1.10.1. Classification

`DecisionTreeClassifier` is a class capable of performing multi-class classification on a dataset.

As with other classifiers, `DecisionTreeClassifier` takes as input two arrays: an array `X`, sparse or dense, of shape `(n_samples, n_features)` holding the training samples, and an array `Y` of integer values, shape `(n_samples,)`, holding the class labels for the training samples:

```
>>> from sklearn import tree
>>> X = [[0, 0], [1, 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])
```

In case that there are multiple classes with the same and highest probability, the classifier will predict the class with the lowest index amongst those classes.

As an alternative to outputting a specific class, the probability of each class can be predicted, which is the fraction of training samples of the 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, ..., K-1]`) classification.

Using the Iris dataset, we can construct a tree as follows:

□ Study

- <https://github.com/amueller/odscon-2015>
- https://www.youtube.com/watch?v=INFm99ss4Hc&ab_channel=OpenDataScience

The image displays two screenshots from the GitHub website. The left screenshot shows the profile of Andreas Mueller (amueller), a Scikit-learn core-developer and Principal Research SDE at Microsoft. His profile includes a bio, a 'Follow' button, and a list of pinned repositories: 'introduction_to_ml_with_python' (Public), 'ml-workshop-1-of-4' (Public), and 'ml-workshop-3-of-4' (Public). Below the pinned repositories is a calendar showing 122 contributions in the last year. The right screenshot shows the repository 'odscon-2015' by amueller. The repository is public and contains a list of files, including 'data', 'executed', '.gitignore', 'LICENSE', and several notebooks for the Open Data Science conference, such as 'Part 01 - Introduction to Scikit-learn.ipynb', 'Part 02 - Unsupervised Transformers.ipynb', and 'Part 03 - Cross-validation.ipynb'.

□ scikit-learn 주요 모듈

분류	모듈 명	설명
예제 데이터	sklearn.datasets	사이킷런에 내장되어 예제로 제공하는 데이터 세트
피처처리	sklearn.preprocessing	데이터 전처리에 필요한 다양한 가공 기능 제공(문자열을 숫자형 코드 값으로 인코딩, 정규화, 스케일링 등)
	sklearn.feature_selection	알고리즘에 큰 영향을 미치는 피처를 우선순위대로 선택션 작업을 수행하는 다양한 기능 제공
	sklearn.feature_extraction	텍스트 데이터나 이미지 데이터의 벡터화된 피처를 추출하는데 사용됨. 예를 들어 텍스트 데이터에서 Count Vectorizer나 Tf-Idf Vectorizer 등을 생성하는 기능 제공. 텍스트 데이터의 피처 추출은 sklearn.feature_extraction.text 모듈에, 이미지 데이터의 피처 추출은 sklearn.feature_extraction.image 모듈에 지원 API가 있음
피처 처리 & 차원 축소	sklearn.decomposition	차원 축소와 관련한 알고리즘을 지원하는 모듈이다. PCA, NMF, Truncated SVD 등을 통해 차원 축소 기능을 수행할 수 있다.
데이터 분리, 검증 & 파라미터 튜닝	sklearn.model_selection	교차 검증을 위한 학습용/테스트용 분리, 그리드 서치(Grid Search)로 최적 파라미터 추출 등의 API 제공
평가	sklearn.metrics	분류, 회귀, 클러스터링, 페어와이즈(Pairwise)에 대한 다양한 성능 측정 방법 제공 Accuracy, Precision, Recall, ROC-AUC, RMSE 등 제공
ML 알고리즘	sklearn.ensemble	앙상블 알고리즘 제공. 랜덤 포레스트, 에이다 부스트, 그래디언트 부스팅 등을 제공
	sklearn.linear_model	주로 선형 회귀, 릿지(Ridge), 라쏘(Lasso) 및 로지스틱 회귀 등 회귀 관련 알고리즘을 지원. 또한 SGD(Stochastic Gradient Descent) 관련 알고리즘도 제공
	sklearn.naïve_bayes	나이브 베이즈 알고리즘 제공. 가우시안 NB. 다항 분포 NB 등
	sklearn.neighbors	최근접 이웃 알고리즘 제공. K-NN(K-Nearest Neighborhood) 등
	sklearn.svm	서포트 벡터 머신 알고리즘 제공
	sklearn.tree	의사 결정 트리 알고리즘 제공
	sklearn.cluster	비지도 클러스터링 알고리즘 제공 (K-평균, 계층형, DBSCAN 등)
유틸리티	sklearn.pipeline	피처 처리 등의 변환과 ML 알고리즘 학습, 예측 등을 함께 묶어서 실행할 수 있는 유틸리티 제공

□ 내장된 예제 데이터 세트

- <https://scikit-learn.org/stable/datasets.html>

The screenshot displays the scikit-learn website's datasets page. The header includes the scikit-learn logo and navigation links: Install, User Guide, API, Examples, Community, and More. A search bar is located in the top right corner. The left sidebar contains a 'Prev' button, 'Up' and 'Next' buttons, a version selector for 'scikit-learn 1.4.0' with a link to 'Other versions', a citation notice, and a 'User Guide' section with a list of topics from 1 to 11. The main content area is titled '7.1. Toy datasets' and lists six datasets: Iris plants, Diabetes, Optical recognition of handwritten digits, Linnerrud, Wine recognition, and Breast cancer wisconsin (diagnostic). Below this is '7.2. Real world datasets' with seven datasets: The Olivetti faces, The 20 newsgroups text, The Labeled Faces in the Wild face recognition, Forest covertypes, RCV1, Kddcup 99, and California Housing. Next is '7.3. Generated datasets' with four generators: for classification and clustering, for regression, for manifold learning, and for decomposition. Finally, '7.4. Loading other datasets' lists four methods: Sample images, Datasets in svmlight / libsvm format, Downloading datasets from the openml.org repository, and Loading from external datasets. A 'Toggle Menu' button is at the bottom left, and the footer contains the copyright notice '© 2007 - 2024, scikit-learn developers (BSD License)' and a link to 'Show this page source'.

scikit-learn
Install User Guide API Examples Community More

Prev Up Next

scikit-learn 1.4.0
Other versions

Please cite us if you use the software.

User Guide

1. Supervised learning
2. Unsupervised learning
3. Model selection and evaluation
4. Inspection
5. Visualizations
6. Dataset transformations
7. Dataset loading utilities
 - 7.1. Toy datasets
 - 7.2. Real world datasets
 - 7.3. Generated datasets
 - 7.4. Loading other datasets
8. Computing with scikit-learn
9. Model persistence
10. Common pitfalls and recommended practices
11. Dispatching

7.1. Toy datasets

- 7.1.1. Iris plants dataset
- 7.1.2. Diabetes dataset
- 7.1.3. Optical recognition of handwritten digits dataset
- 7.1.4. Linnerrud dataset
- 7.1.5. Wine recognition dataset
- 7.1.6. Breast cancer wisconsin (diagnostic) dataset

7.2. Real world datasets

- 7.2.1. The Olivetti faces dataset
- 7.2.2. The 20 newsgroups text dataset
- 7.2.3. The Labeled Faces in the Wild face recognition dataset
- 7.2.4. Forest covertypes
- 7.2.5. RCV1 dataset
- 7.2.6. Kddcup 99 dataset
- 7.2.7. California Housing dataset

7.3. Generated datasets

- 7.3.1. Generators for classification and clustering
- 7.3.2. Generators for regression
- 7.3.3. Generators for manifold learning
- 7.3.4. Generators for decomposition

7.4. Loading other datasets

- 7.4.1. Sample images
- 7.4.2. Datasets in svmlight / libsvm format
- 7.4.3. Downloading datasets from the openml.org repository
- 7.4.4. Loading from external datasets

Toggle Menu

© 2007 - 2024, scikit-learn developers (BSD License). [Show this page source](#)

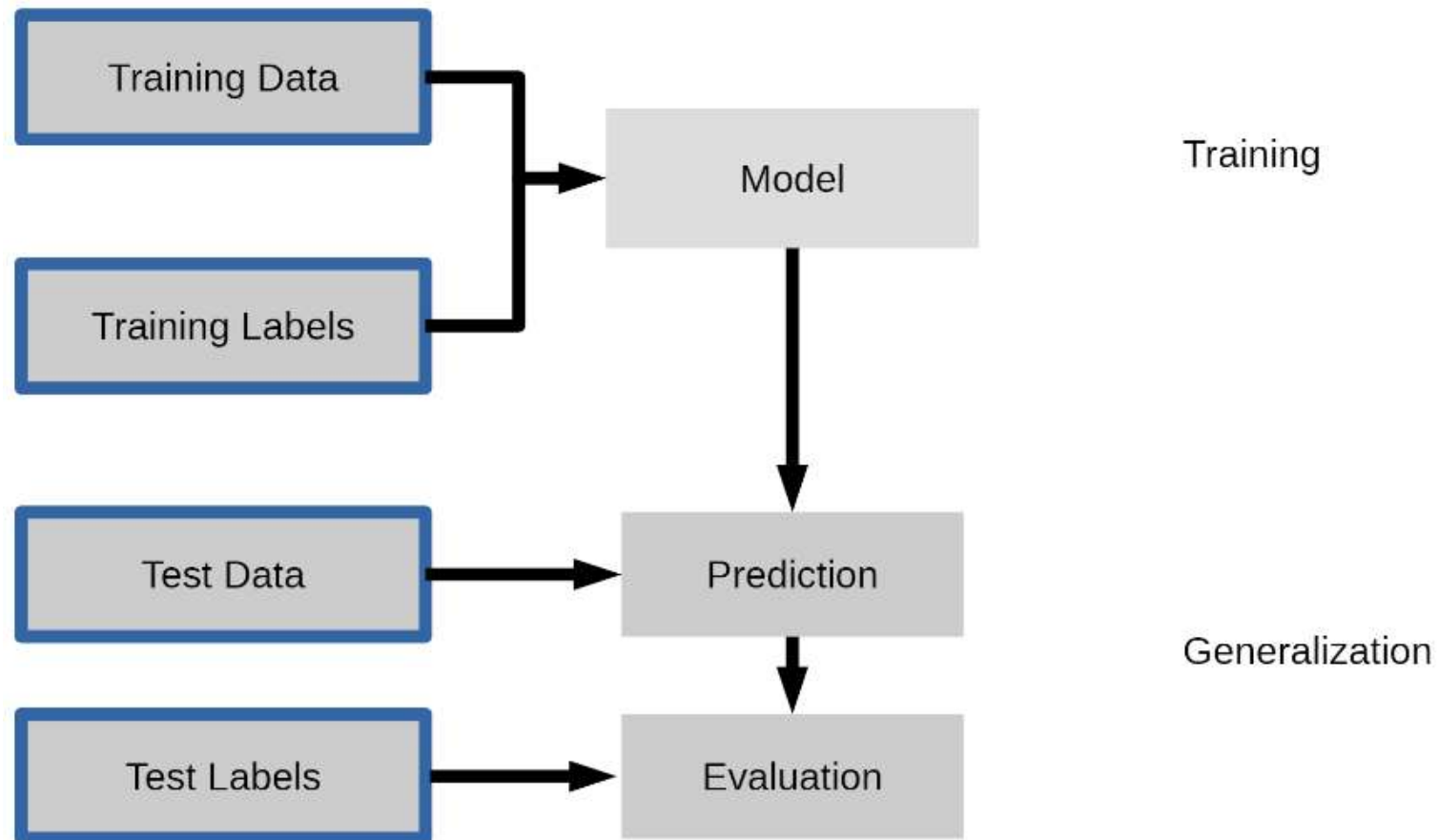
□ Model Selection

The screenshot shows the scikit-learn documentation page for the `sklearn.model_selection.train_test_split` function. The page layout includes a top navigation bar with links for 'Install', 'User Guide', 'API', 'Examples', 'Community', and 'More'. A search bar is located on the right. On the left side, there are navigation buttons for 'Prev', 'Up', and 'Next', along with a version selector for 'scikit-learn 1.4.0' and a 'Toggle Menu' button at the bottom. The main content area features the function name in a large blue header, followed by the function signature: `sklearn.model_selection.train_test_split(*arrays, test_size=None, train_size=None, random_state=None, shuffle=True, stratify=None)`. Below this, there is a brief description: 'Split arrays or matrices into random train and test subsets.' and a more detailed explanation: 'Quick utility that wraps input validation, `next(ShuffleSplit()).split(X, y)`, and application to input data into a single call for splitting (and optionally subsampling) data into a one-liner.' A link to 'Read more in the User Guide.' is provided. The 'Parameters' section lists the following:

- *arrays** : *sequence of indexables with same length / shape[0]*
Allowed inputs are lists, numpy arrays, scipy-sparse matrices or pandas dataframes.
- test_size** : *float or int, default=None*
If float, should be between 0.0 and 1.0 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. If `train_size` is also None, it will be set to 0.25.
- train_size** : *float or int, default=None*
If float, should be between 0.0 and 1.0 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.
- random_state** : *int, RandomState instance or None, default=None*
Controls the shuffling applied to the data before applying the split. Pass an int for reproducible output across multiple function calls. See [Glossary](#).
- shuffle** : *bool, default=True*
Whether or not to shuffle the data before splitting. If `shuffle=False` then `stratify` must be None.
- stratify** : *array-like, default=None*
If not None, data is split in a stratified fashion, using this as the class labels. Read more in the User Guide.

※ 출처 : https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html#sklearn.model_selection.train_test_split

□ Supervised Machine Learning



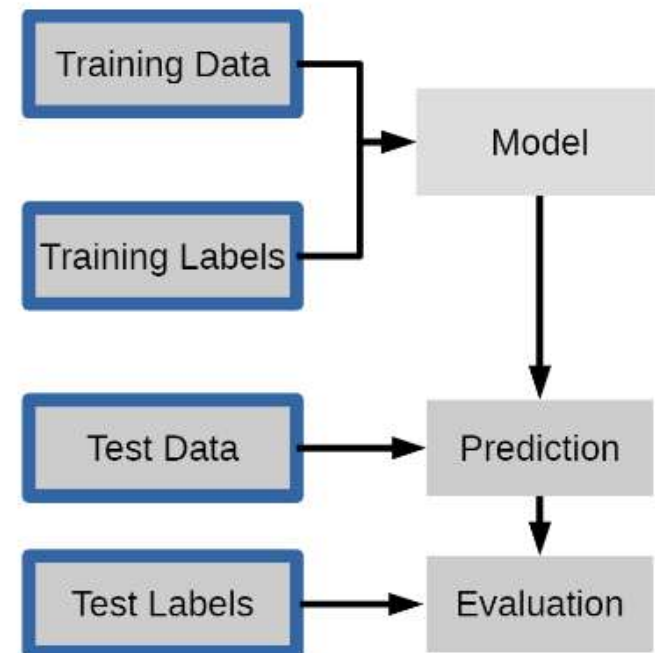
□ Supervised Machine Learning

```
clf = RandomForestClassifier()
```

```
clf.fit(X_train, y_train)
```

```
y_pred = clf.predict(X_test)
```

```
clf.score(X_test, y_test)
```



□ Basic API of Scikit-Learn

`estimator.fit(X, [y])`

`estimator.predict`

`estimator.transform`

Classification

Preprocessing

Regression

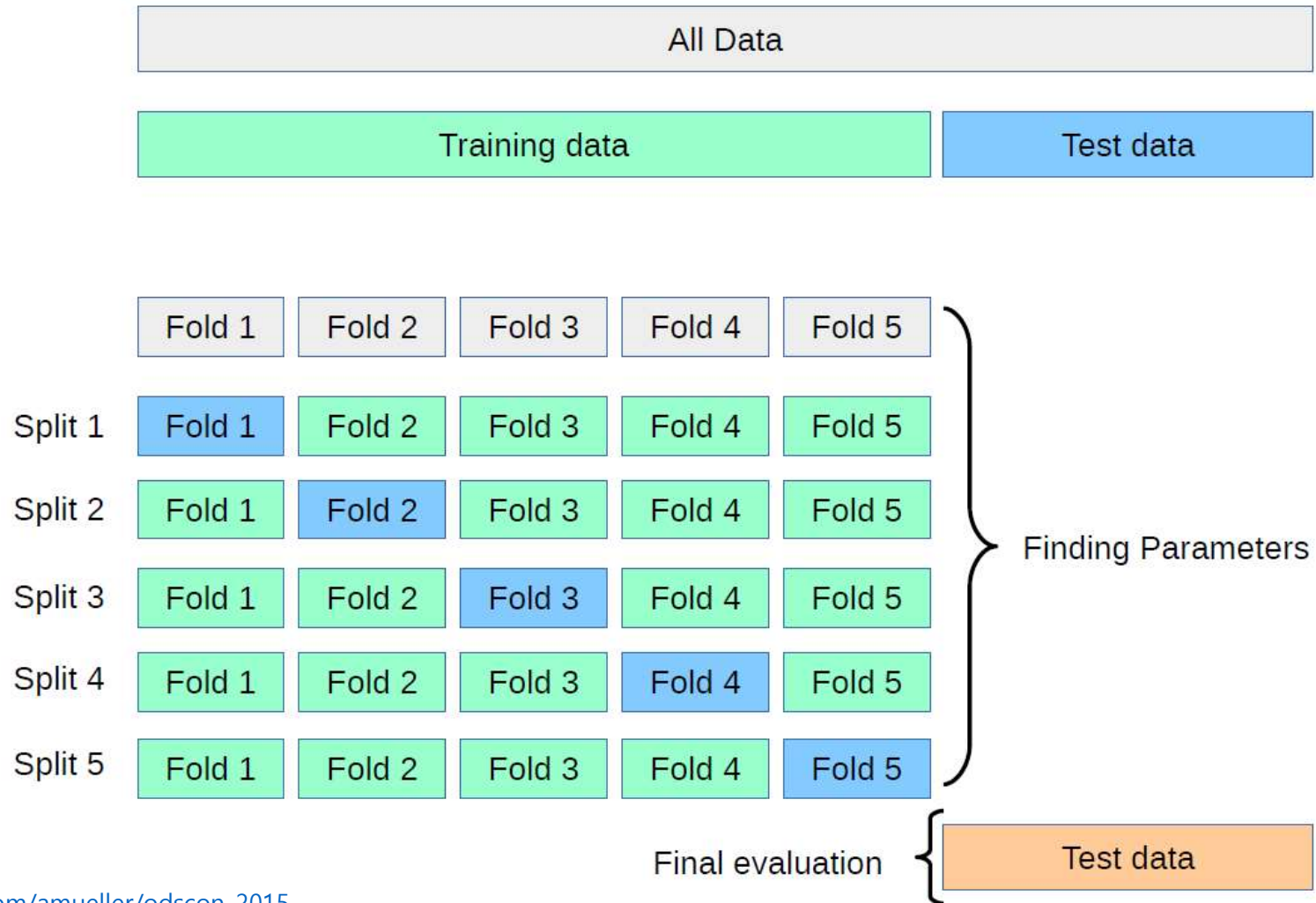
Dimensionality reduction

Clustering

Feature selection

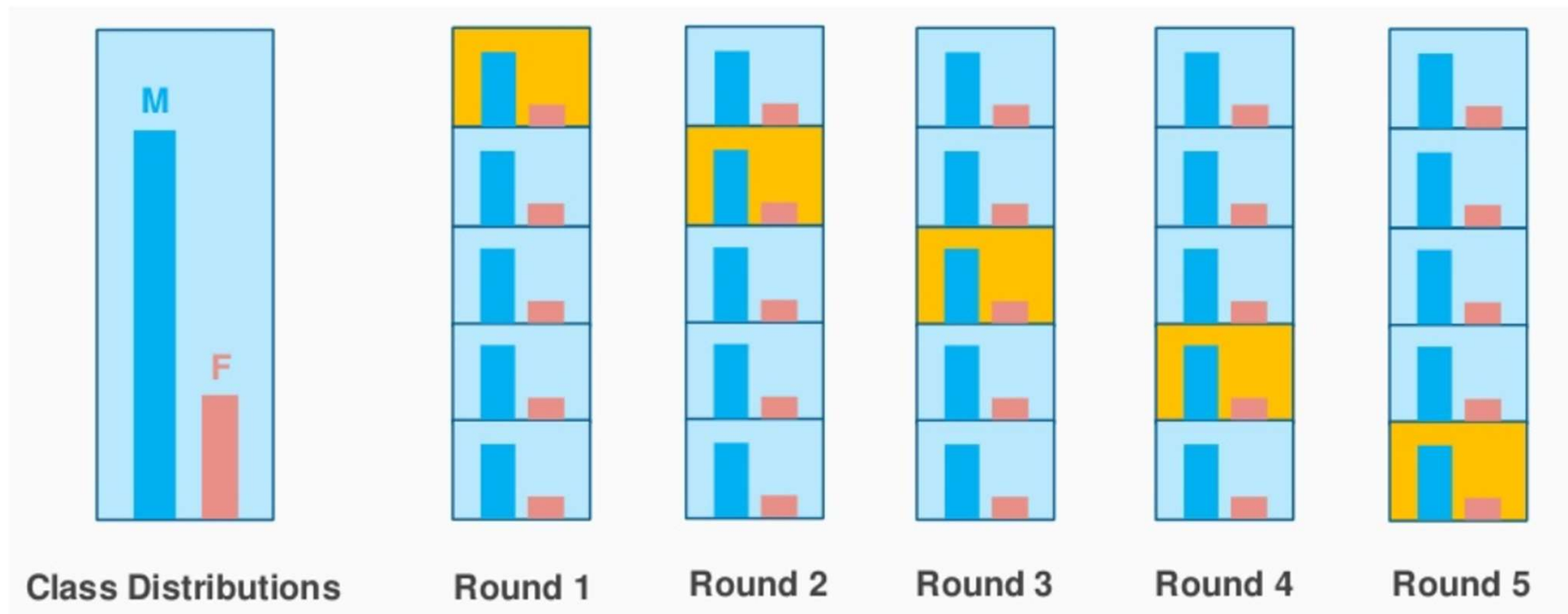
Feature extraction

□ 교차 검증(Cross Validation) : K-Fold



※ 출처 : <https://github.com/amueller/odscon-2015>

□ 교차 검증(Cross Validation) : Stratified K-Fold



※ 출처 : <https://stats.stackexchange.com/questions/49540/understanding-stratified-cross-validation>

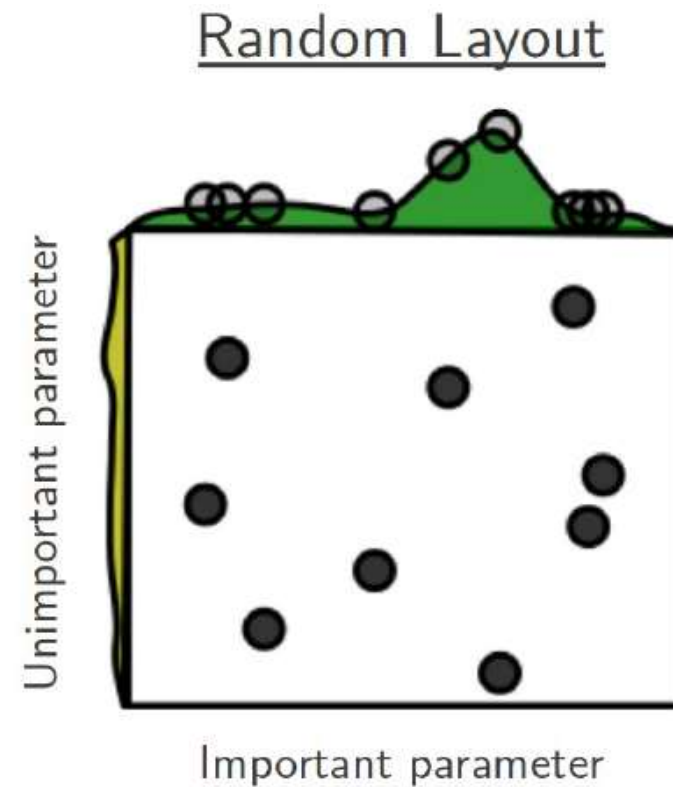
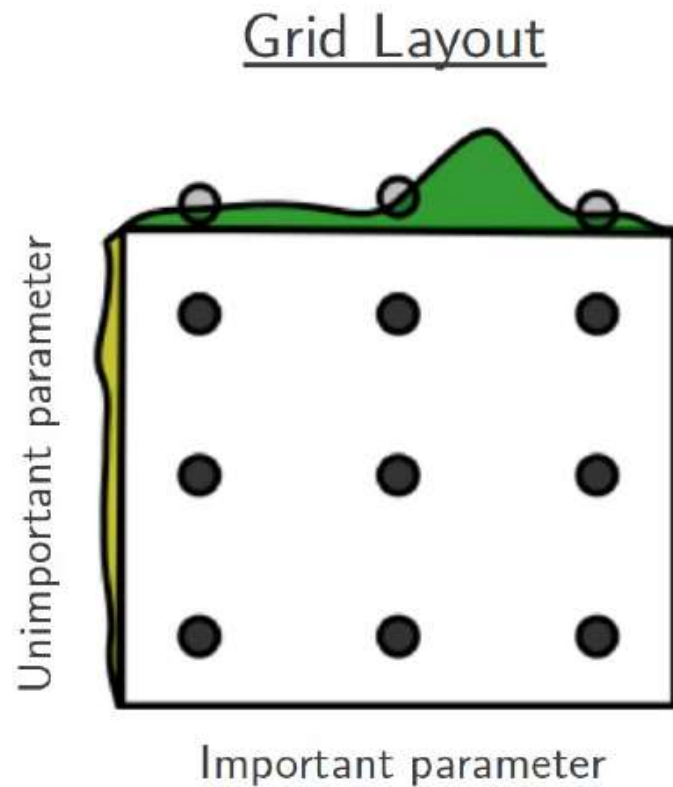
□ 교차 검증(Cross Validation) : cross_val_score()

The screenshot shows the scikit-learn documentation page for `sklearn.model_selection.cross_val_score`. The page has a light blue header with the scikit-learn logo and navigation links: Install, User Guide, API, Examples, Community, and More. Below the header, there are buttons for Prev, Up, and Next. A pink box indicates the current version is scikit-learn 1.4.0, with a link to Other versions. A yellow box asks users to cite the software. The left sidebar contains a list of links: `sklearn.model_selection.cross_val_score`, `cross_val_score`, Examples using, and `sklearn.model_selection.cross_val`. The main content area has a blue header with the function name. Below it, the function signature is shown: `sklearn.model_selection.cross_val_score(estimator, X, y=None, *, groups=None, scoring=None, cv=None, n_jobs=None, verbose=0, fit_params=None, params=None, pre_dispatch='2*n_jobs', error_score=nan)` with a [source] link. The text "Evaluate a score by cross-validation." and "Read more in the User Guide." are present. The Parameters section lists: **estimator**: estimator object implementing 'fit' (The object to use to fit the data.); **X**: {array-like, sparse matrix} of shape (n_samples, n_features) (The data to fit. Can be for example a list, or an array.); **y**: array-like of shape (n_samples,) or (n_samples, n_outputs), default=None (The target variable to try to predict in the case of supervised learning.); **groups**: array-like of shape (n_samples,), default=None (Group labels for the samples used while splitting the dataset into train/test set. Only used in conjunction with a "Group" cv instance (e.g., `GroupKFold`)). A yellow box contains a note: "Changed in version 1.4: groups can only be passed if metadata routing is not enabled via `sklearn.set_config(enable_metadata_routing=True)`. When routing is enabled, pass groups alongside other metadata via the params argument instead. E.g.: `cross_val_score(..., params={'groups': groups})`." **scoring**: str or callable, default=None (A str (see model evaluation documentation) or a scorer callable object / function with signature `scorer(estimator, X, y)` which should return only a single value. Similar to `cross_validate` but only a single metric is permitted. If None, the estimator's default scorer (if available) is used.

※ 출처 : https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html

□ Parameter Search

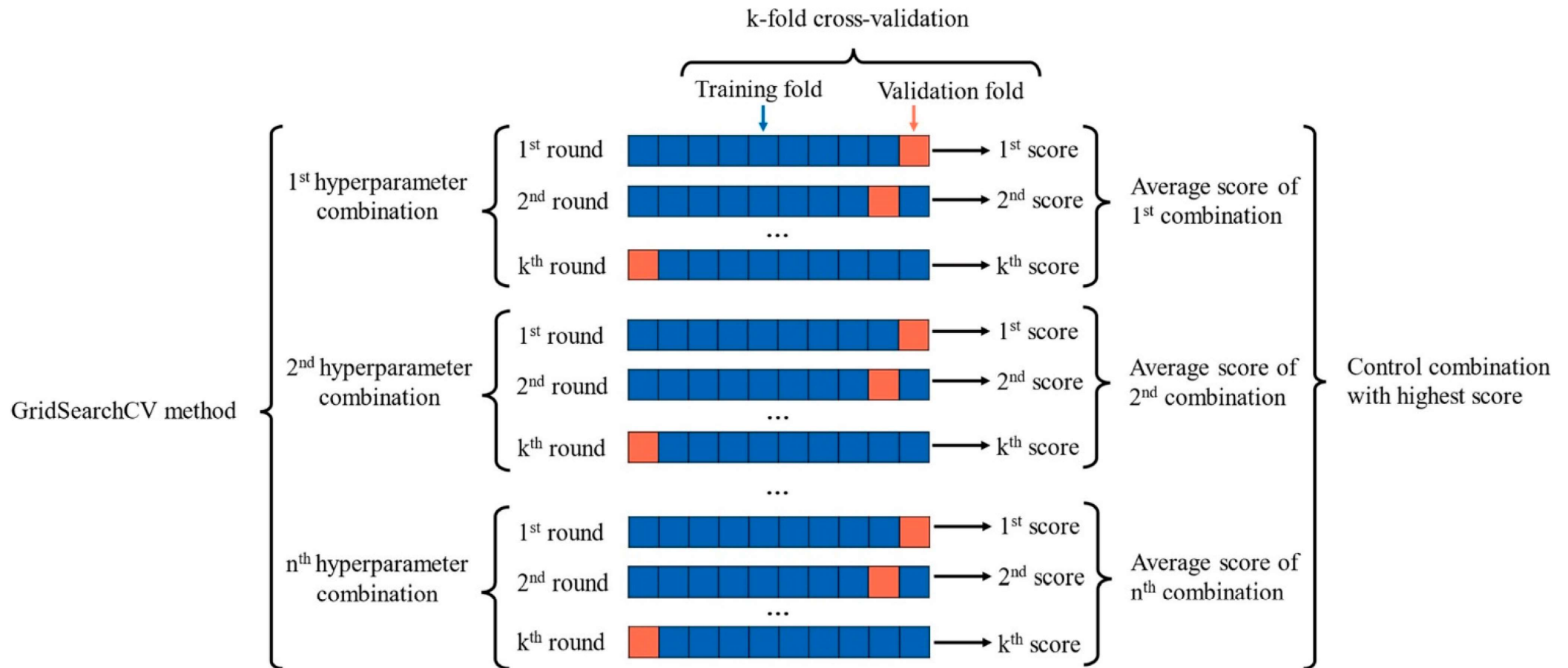
- Grid vs. Random



Source: Bergstra and Bengio

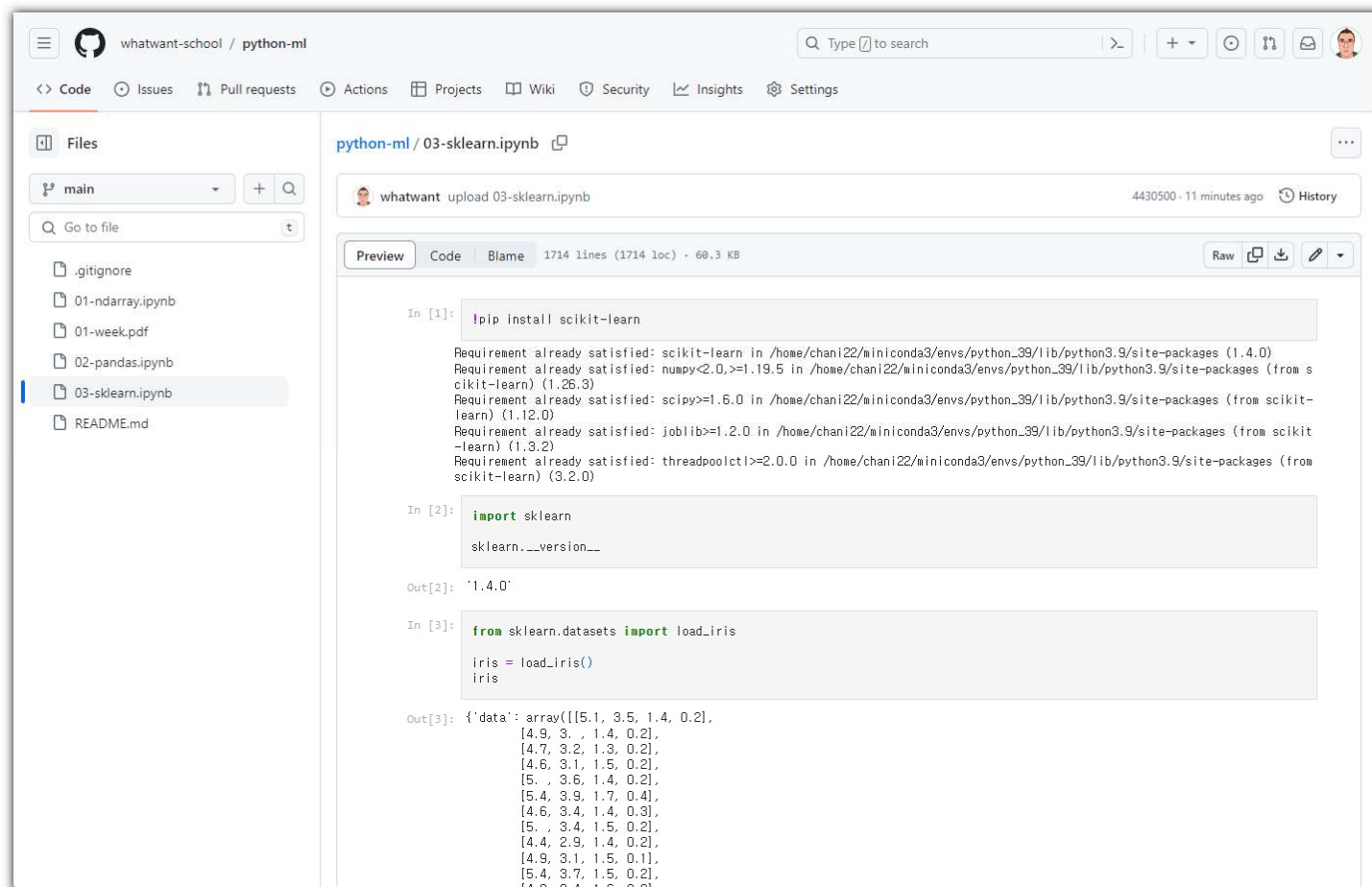
※ 출처 : <https://github.com/amueller/odscon-2015>

□ 교차 검증(Cross Validation) : GridSearchCV



❑ Source Code

- <https://github.com/whatwant-school/python-ml/blob/main/03-sklearn.ipynb>



You've really worked hard today

Next Week ~ ?

02

사이킷런으로 시작하는 머신러닝

01. 사이킷런 소개와 특징	85
02. 첫 번째 머신러닝 만들어 보기 - 붓꽃 품종 예측하기	86
03. 사이킷런의 기본 프레임워크 익히기	91
Estimator 이해 및 fit(), predict() 메서드	91
사이킷런의 주요 모듈	92
내장된 예제 데이터 세트	94
04. Model Selection 모듈 소개	98
학습/테스트 데이터 세트 분리 - train_test_split()	98
교차 검증	100
GridSearchCV - 교차 검증과 최적 하이퍼 파라미터 튜닝을 한 번에	111
05. 데이터 전처리	116
데이터 인코딩	116
피처 스케일링과 정규화	122
StandardScaler	123
MinMaxScaler	125
학습 데이터와 테스트 데이터의 스케일링 변환 시 유의점	126
06. 사이킷런으로 수행하는 타이타닉 생존자 예측	129
07. 정리	143
01. 정확도(Accuracy)	146
02. 오차 행렬	150
03. 정밀도와 재현율	154
정밀도/재현율 트레이드오프	157
정밀도와 재현율의 명점	165

03

평가

04. F1 스코어	166
05. ROC 곡선과 AUC	167
06. 피마 인디언 당뇨병 예측	172
07. 정리	180

04

분류

01. 분류(Classification)의 개요	181
02. 결정 트리	183
결정 트리 모델의 특징	185
결정 트리 파라미터	186
결정 트리 모델의 시각화	187
결정 트리 과적합(Overfitting)	198
결정 트리 실습 - 사용자 행동 인식 데이터 세트	200
03. 앙상블 학습	210
앙상블 학습 개요	210
보팅 유형 - 하드 보팅(Hard Voting)과 소프트 보팅(Soft Voting)	212
보팅 분류기(Voting Classifier)	213
04. 랜덤 포레스트	216
랜덤 포레스트의 개요 및 실습	216
랜덤 포레스트 하이퍼 파라미터 및 튜닝	218
GBM의 개요 및 실습	221
05. GBM(Gradient Boosting Machine)	221
GBM 하이퍼 파라미터 소개	224
XGBoost 개요	225

Who ~ ?

See you Next Weekend ~ ?