scikit.learn - Machine Learning with Python

-
-
-

Machine Learning with Python

MachineLearning

Machine Learning is a discipline involving algorithms designed to find patterns in and make predictions about data. It is nearly ubiquitous in our world today, and used in everything from web searches to financial forecasts to studies of the nature of the Universe. This tutorial will offer an introduction to scikit-learn, a python machine learning package, and to the central concepts of Machine Learning.

MachineLearning We will introduce the basic categories of learning problems and how to implement them using scikit-learn. From this foundation, we will explore practical examples of machine learning using real-world data, from handwriting analysis to automated classification of astronomical images.

Machine Learning with Python

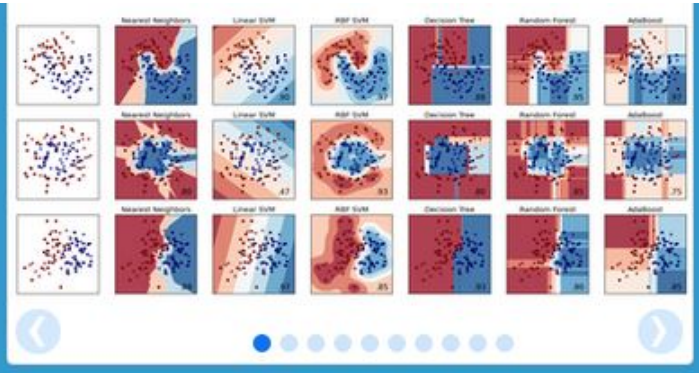
MachineLearning

Machine Learning is a discipline involving algorithms designed to find patterns in and make predictions about data. It is nearly ubiquitous in our world today, and used in everything from web searches to financial forecasts to studies of the nature of the Universe. This tutorial will offer an introduction to scikit-learn, a python machine learning package, and to the central concepts of Machine Learning.

MachineLearning We will introduce the basic categories of learning problems and how to implement them using scikit-learn. From this foundation, we will explore practical examples of machine learning using real-world data, from handwriting analysis to automated classification of astronomical images.

Getting ready The datasets in scikit-learn are contained within the datasets module. Use the following command to import these datasets:

```
>>> from sklearn import datasets
>>> import numpy as np
```



scikit-learn

Machine Learning in Python

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

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: *SVM, nearest neighbors, random forest, ...* — *Examples*

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: *SVR, ridge regression, Lasso, ...* — *Examples*

Clustering

Automatic grouping of similar objects in

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: *k-Means, spectral clusteri mean-shift, ...* — E

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: *PCA, feature selection, non-negative matrix factorization.* — *Examples*

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning
Modules: *grid search, cross validation, metrics.* — *Examples*

Preprocessing

Feature extraction and normalization.

Application: Transforming input data s text for use with machine learning algor
Modules: *preprocessing, feature extra* — E

Classification

- **Description:** Identifying to which category an object belongs to.
- **Applications:** Spam detection, Image recognition.
- **Algorithms:** SVM, nearest neighbors, random forest,

Regression

- **Description:** Predicting a continuous-valued attribute associated with an object.
- **Applications:** Drug response, Stock prices.
- **Algorithms:** SVR, ridge regression, Lasso,

Clustering

Automatic grouping of similar objects into sets. Applications: Customer segmentation, Grouping experiment outcomes Algorithms: k-Means, spectral clustering, mean-shift, ...

Dimensionality Reduction

- **Description:** Reducing the number of random variables to consider.
- **Applications:** Visualization, Increased efficiency
- **Algorithms:** PCA, feature selection, non-negative matrix factorization.

Model selection

- **Description:** Comparing, validating and choosing parameters and models.
- **Goal:** Improved accuracy via parameter tuning
- **Modules:** grid search, cross validation, metrics

Preprocessing

- **Description:** Feature extraction and normalization.
- **Application:** Transforming input data such as text for use with machine learning algorithms.
- **Modules:** preprocessing, feature extraction.

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.

Decision Trees

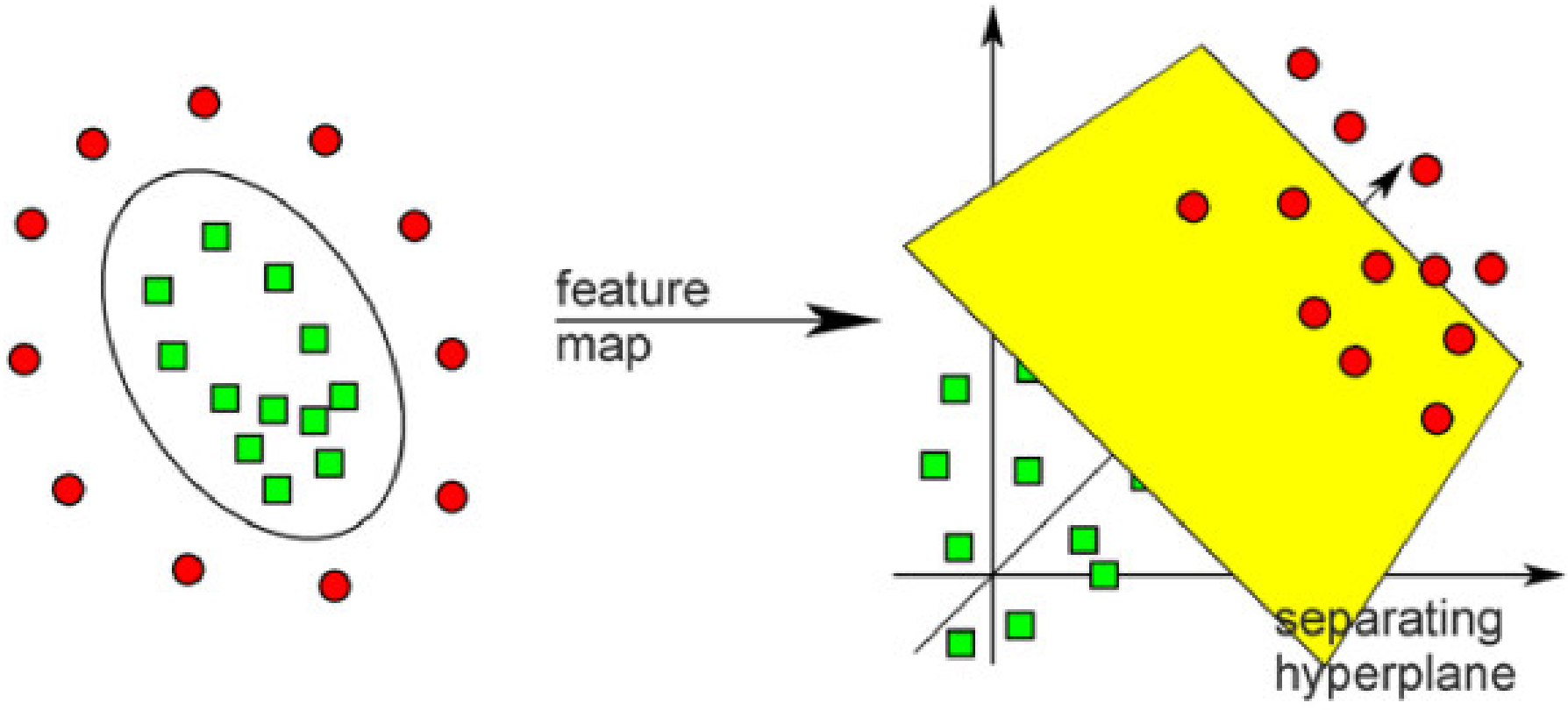
- For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of *If-Then-Else* decision rules.
- The deeper the tree, the more complex the decision rules and the fitter the model.

Decision Trees

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)
```

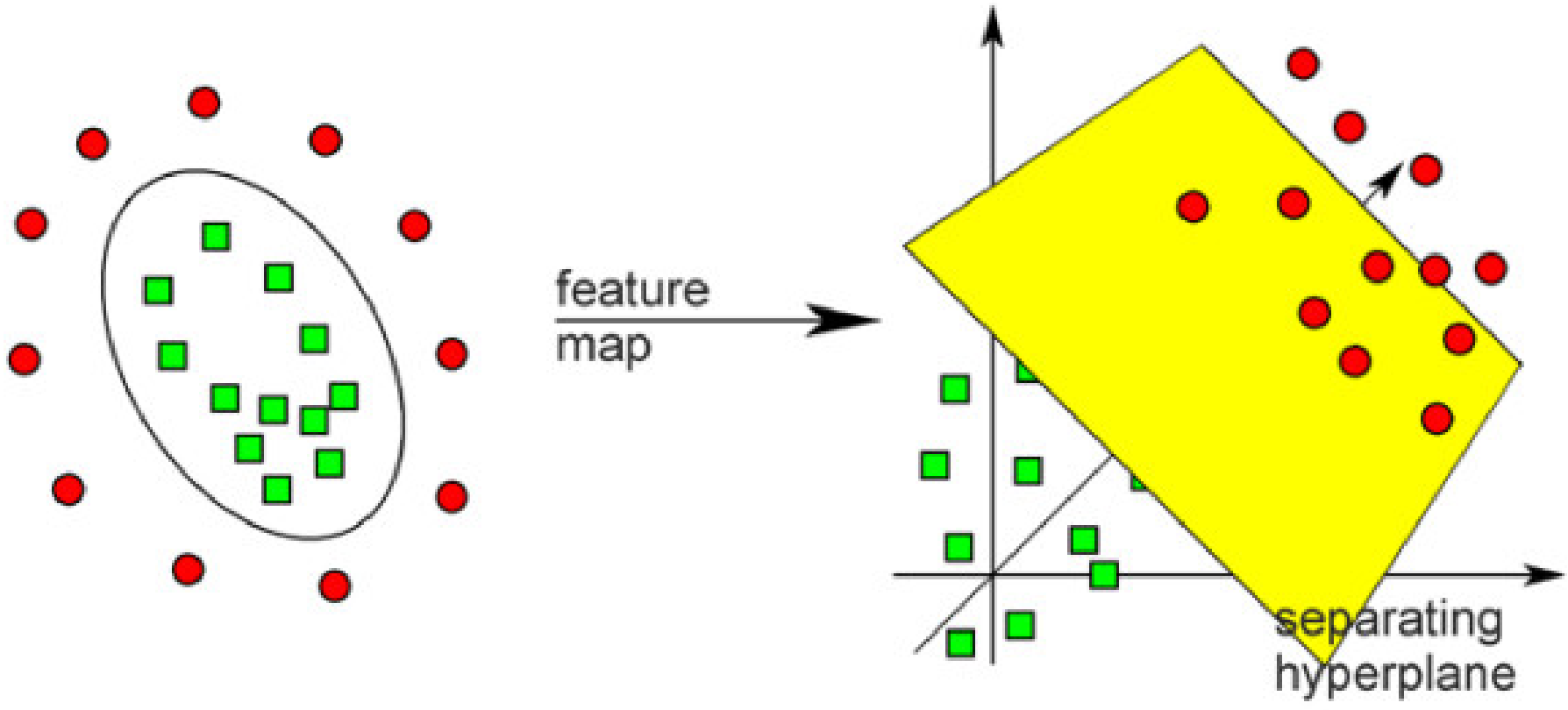
Separation may be easier in higher dimensions



complex in low dimensions

simple in higher dimensions

Separation may be easier in higher dimensions



complex in low dimensions

simple in higher dimensions

scikit.learn - Machine Learning with Python

```
import scikit.learn
```