Scikit library

Fitting and predicting: estimator basics

Scikit-learn provides dozens of built-in machine learning algorithms and models, called [estimators](https://scikit-learn.org/stable/glossary.html#term-estimators). Each estimator can be fitted to some data using its [fit](https://scikit-learn.org/stable/glossary.html#term-fit) method.

Transformers and pre-processors

Machine learning workflows are often composed of different parts. A typical pipeline consists of a pre-processing step that transforms or imputes the data, and a final predictor that predicts target values.

Pipelines: chaining pre-processors and estimators[¶](https://scikit-learn.org/stable/getting_started.html#pipelines-chaining-pre-processors-and-estimators)

Transformers and estimators (predictors) can be combined together into a single unifying object: a [**Pipeline**](https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html#sklearn.pipeline.Pipeline). The pipeline offers the same API as a regular estimator: it can be fitted and used for prediction with fit and predict.

Preprocessing

StandardScaler

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

**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 matrix, a copy may still be returned.

**with\_mean *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.

**with\_std *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 with\_std=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 with\_mean=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 with\_std=False.

**n\_samples\_seen\_ *int or array, shape (n\_features,)***

The number of samples processed by the estimator for each feature. If there are not missing samples, the n\_samples\_seen will be an integer, otherwise it will be an array. Will be reset on new calls to fit, but increments across partial\_fit calls.

Methods of Standard Scaler

[**fit**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler.fit)(self, X[, y])

Compute the mean and std to be used for later scaling.

[**fit\_transform**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler.fit_transform)(self, X[, y])

Fit to data, then transform it.

[**get\_params**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler.get_params)(self[, deep])

Get parameters for this estimator.

[**inverse\_transform**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler.inverse_transform)(self, X[, copy])

Scale back the data to the original representation

[**partial\_fit**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler.partial_fit)(self, X[, y])

Online computation of mean and std on X for later scaling.

[**set\_params**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler.set_params)(self, \\*\\*params)

Set the parameters of this estimator.

[**transform**](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler.transform)(self, X[, copy])

Perform standardization by centering and scaling

Random Forests

In random forests, each tree in the ensemble is built from a sample drawn with replacement (i.e., 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 max\_features.

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.

### Parameters

n\_estimators

Number of trees in the forest. The larger the better, but also the longer it will take to compute.

n\_features

Number of features in the data/

max\_features

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 max\_features=None (always considering all features instead of a random subset) for regression problems, and max\_features="sqrt" (using a random subset of size sqrt(n\_features)) for classification

bootstrap

In random forests, bootstrap samples are used by default (bootstrap=True)

oob\_score

When using bootstrap sampling the generalization accuracy can be estimated on the left out or out-of-bag samples. This can be enabled by setting oob\_score=True.

**Methods**

1. [**apply**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.apply)(clf\_model, X)

Apply trees in the forest to X, return leaf indices

1. [**decision\_path**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.decision_path)(clf\_model, X)

Return the decision path in the forest.

1. [**fit**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.fit)(clf\_model, X, y[, sample\_weight])

Build a forest of trees from the training set (X, y).

1. [**get\_params**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.get_params)(clf\_model [, deep])

Get parameters for this estimator.

1. [**predict**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.predict)(clf\_model, X)

Predict class for X.

1. [**predict\_log\_proba**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.predict_log_proba)(self, X)

Predict class log-probabilities for X.

1. [**predict\_proba**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.predict_proba)(self, X)

Predict class probabilities for X.

1. [**score**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.score)(clf\_model, X, y[, sample\_weight])

Return the mean accuracy on the given test data and labels.

1. [**set\_params**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier.set_params)(clf\_model, \\*\\*params)

Set the parameters of this estimator.