sklearn.model_selection.RepeatedKFold

class `sklearn.model_selection.`**`RepeatedKFold`**(**, n_splits=5, n_repeats=10, random_state=None*) [source]

Repeated K-Fold cross validator.

Repeats K-Fold n times with different randomization in each repetition.

Read more in the User Guide.

Parameters:

n_splits : *int, default=5*
Number of folds. Must be at least 2.

n_repeats : *int, default=10*
Number of times cross-validator needs to be repeated.

random_state : *int or RandomState instance, default=None*
Controls the randomness of each repeated cross-validation instance. Pass an int for reproducible output across multiple function calls. See Glossary.

See also:

`RepeatedStratifiedKFold`
Repeats Stratified K-Fold n times.

Notes

Randomized CV splitters may return different results for each call of split. You can make the results identical by setting `random_state` to an integer.

Examples

```
>>> import numpy as np
>>> from sklearn.model_selection import RepeatedKFold
>>> X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
>>> y = np.array([0, 0, 1, 1])
>>> rkf = RepeatedKFold(n_splits=2, n_repeats=2, random_state=2652124)
>>> for train_index, test_index in rkf.split(X):
...     print("TRAIN:", train_index, "TEST:", test_index)
...     X_train, X_test = X[train_index], X[test_index]
...     y_train, y_test = y[train_index], y[test_index]
...
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

`get_n_splits`(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__`(*self, *, n_splits=5, n_repeats=10, random_state=None*) [source]

Initialize self. See help(type(self)) for accurate signature.

`get_n_splits`(*self, X=None, y=None, groups=None*) [source]

Returns the number of splitting iterations in the cross-validator

Parameters:

X : *object*
Always ignored, exists for compatibility. `np.zeros(n_samples)` may be used as a placeholder.

Toggle Menu ***ct***

Always ignored, exists for compatibility. `np.zeros(n_samples)` may be used as a placeholder.

groups : *array-like of shape (n_samples,), default=None*

Group labels for the samples used while splitting the dataset into train/test set.

Returns:

n_splits : *int*

Returns the number of splitting iterations in the cross-validator.

split(*self*, *X*, *y=None*, *groups=None*) [source]

Generates indices to split data into training and test set.

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 : *array-like of length n_samples*

The target variable for supervised learning problems.

groups : *array-like of shape (n_samples,), default=None*

Group labels for the samples used while splitting the dataset into train/test set.

Yields:

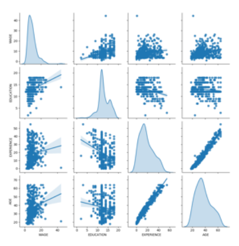
train : *ndarray*

The training set indices for that split.

test : *ndarray*

The testing set indices for that split.

Examples using `sklearn.model_selection.RepeatedKFold` ¶



Common pitfalls in interpretation of coefficients of linear models

Toggle Menu