СПбГЭТУ «ЛЭТИ»
ПЕРВЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ

Юлия Тероева

ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В МЕДИЦИНЕ

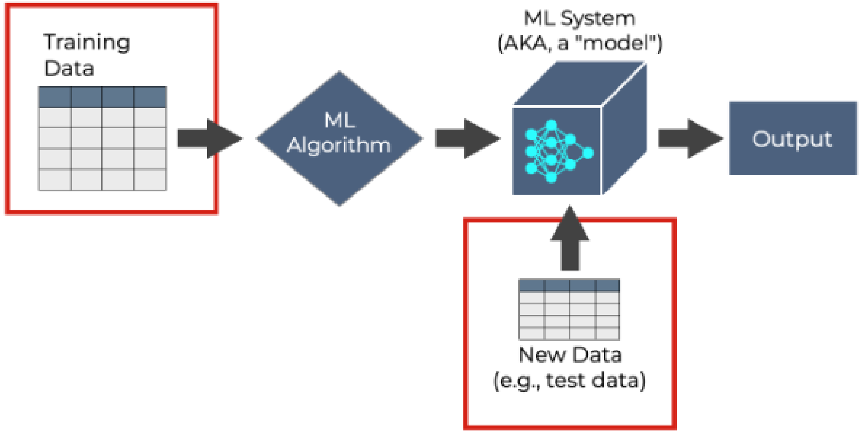
Модуль 2. Методы искусственного интеллекта

Тема 3.2 ИНС на Python

Часть 6. Подготовка обучающей и контрольной выборок

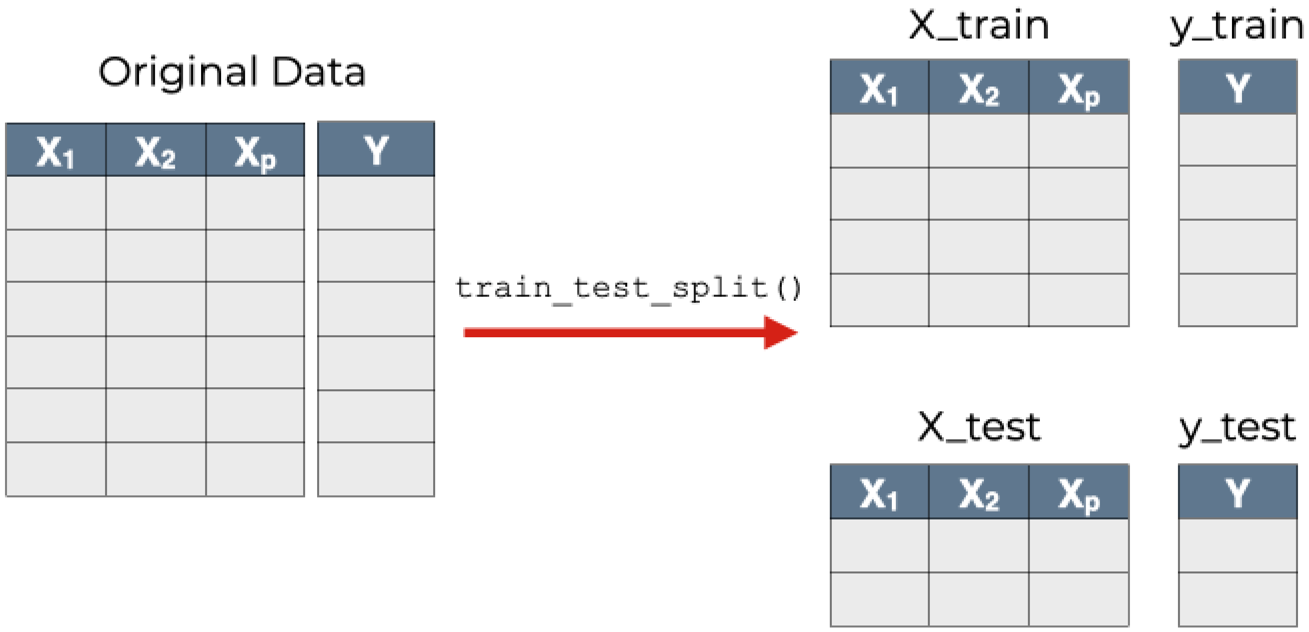
ОБЩИЕ СВЕДЕНИЯ

- **Обучить модель** – значит *подобрать такие коэффициенты* и другие параметры, которые *удовлетворяют условию оптимизации*.
- Обучение осуществляется на большом наборе данных – обучающей выборке (**train dataset**).
- Тестирование эффективности модели проводится на отложенном наборе данных (**test dataset**).

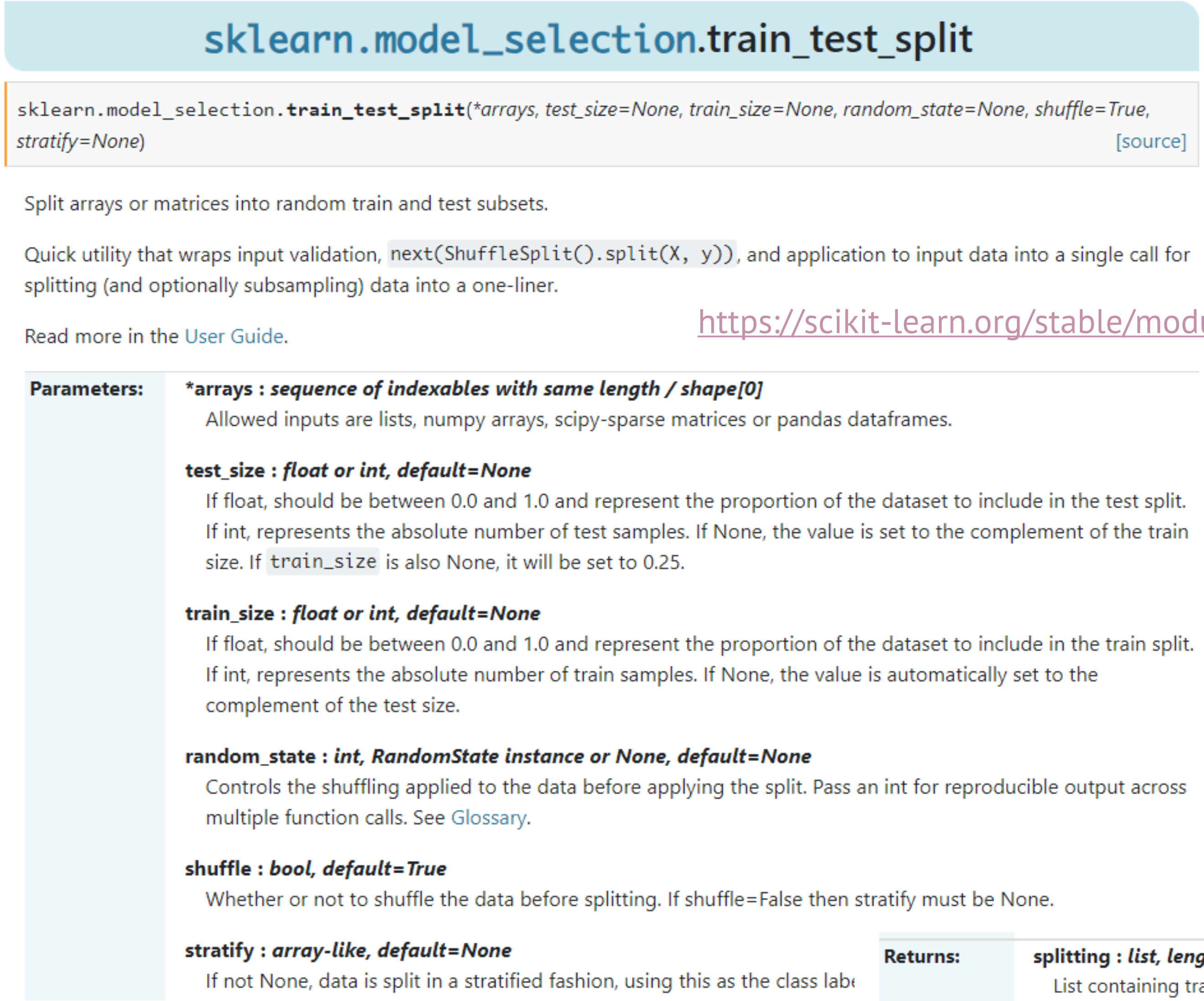


АВТОМАТИЧЕСКОЕ РАЗДЕЛЕНИЕ ТАБЛИЦЫ

- В большинстве задач для разделения выборки на подвыборки удобнее использовать функцию **train_test_split()** из библиотеки sklearn.



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



sklearn.model_selection.train_test_split

`sklearn.model_selection.train_test_split`(**arrays, test_size=None, train_size=None, random_state=None, shuffle=True, stratify=None*) [source]

Split arrays or matrices into random train and test subsets.

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.

Read more in the User Guide.

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

Parameters:

arrays : *sequence of indexables with same length / shape[0]
Allowed inputs are lists, numpy arrays, scipy-sparse matrices or pandas dataframes.

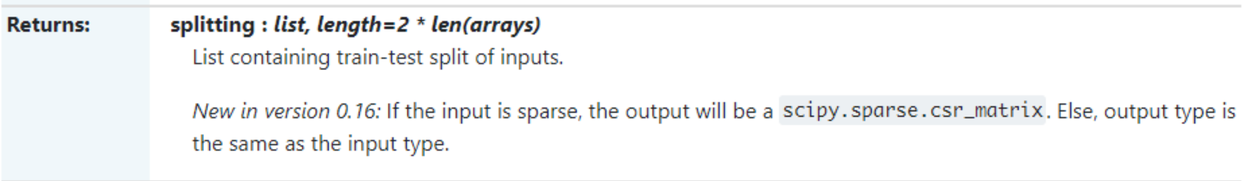
test_size : *float or int, default=None*
If float, should be between 0.0 and 1.0 and represent the proportion of the dataset to include in the test split. If int, represents the absolute number of test samples. If None, the value is set to the complement of the train size. If `train_size` is also None, 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 labe



Returns:

splitting : *list, length=2 * len(arrays)*
List containing train-test split of inputs.

New in version 0.16: If the input is sparse, the output will be a `scipy.sparse.csr_matrix`. Else, output type is the same as the input type.

ПРИМЕНЕНИЕ TRAIN_TEST_SPLIT()

```
import numpy as np
from sklearn.model_selection import train_test_split
X, y = np.arange(10).reshape((5, 2)), range(5)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
y_train, y_test  = train_test_split(y, shuffle=False)
```

ПРИМЕНЕНИЕ TRAIN_TEST_SPLIT()

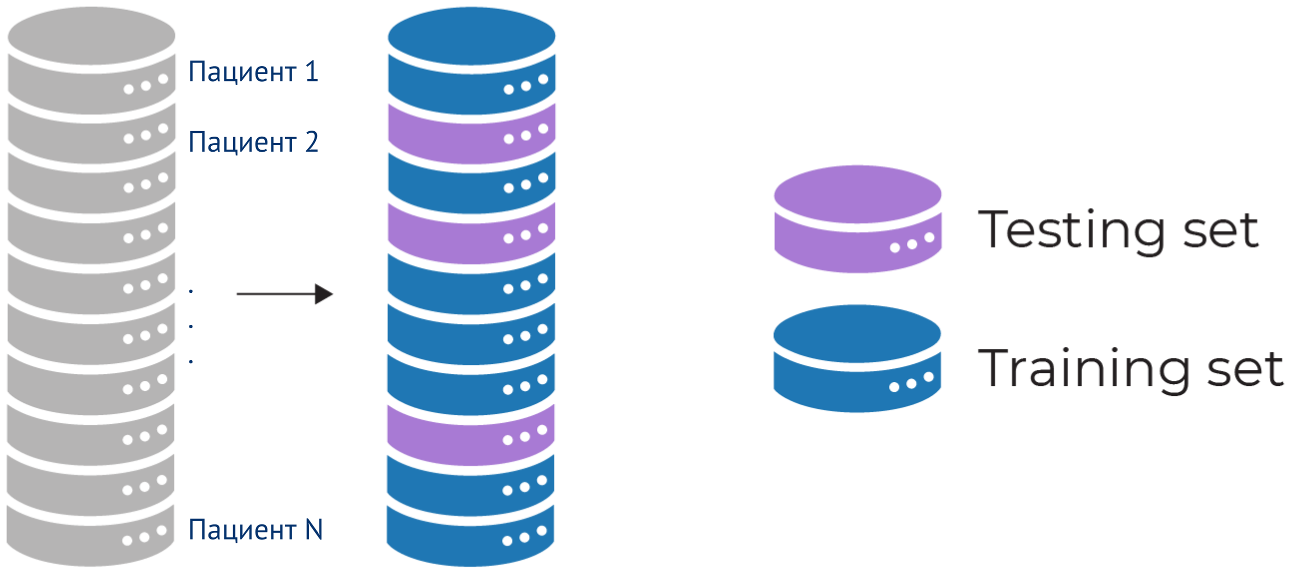
```
import pandas as pd
data = pd.read_csv('heart_failure_clinical_records_dataset.csv')

X = data.drop('DEATH_EVENT', axis=1) # данные, кроме строки меток
Y = data[['DEATH_EVENT']].to_numpy().ravel() # целевая переменная

from sklearn.model_selection import train_test_split
X_train, X_test,Y_train,Y_test = train_test_split(X, Y, random_state=100,test_size=0.2)
```

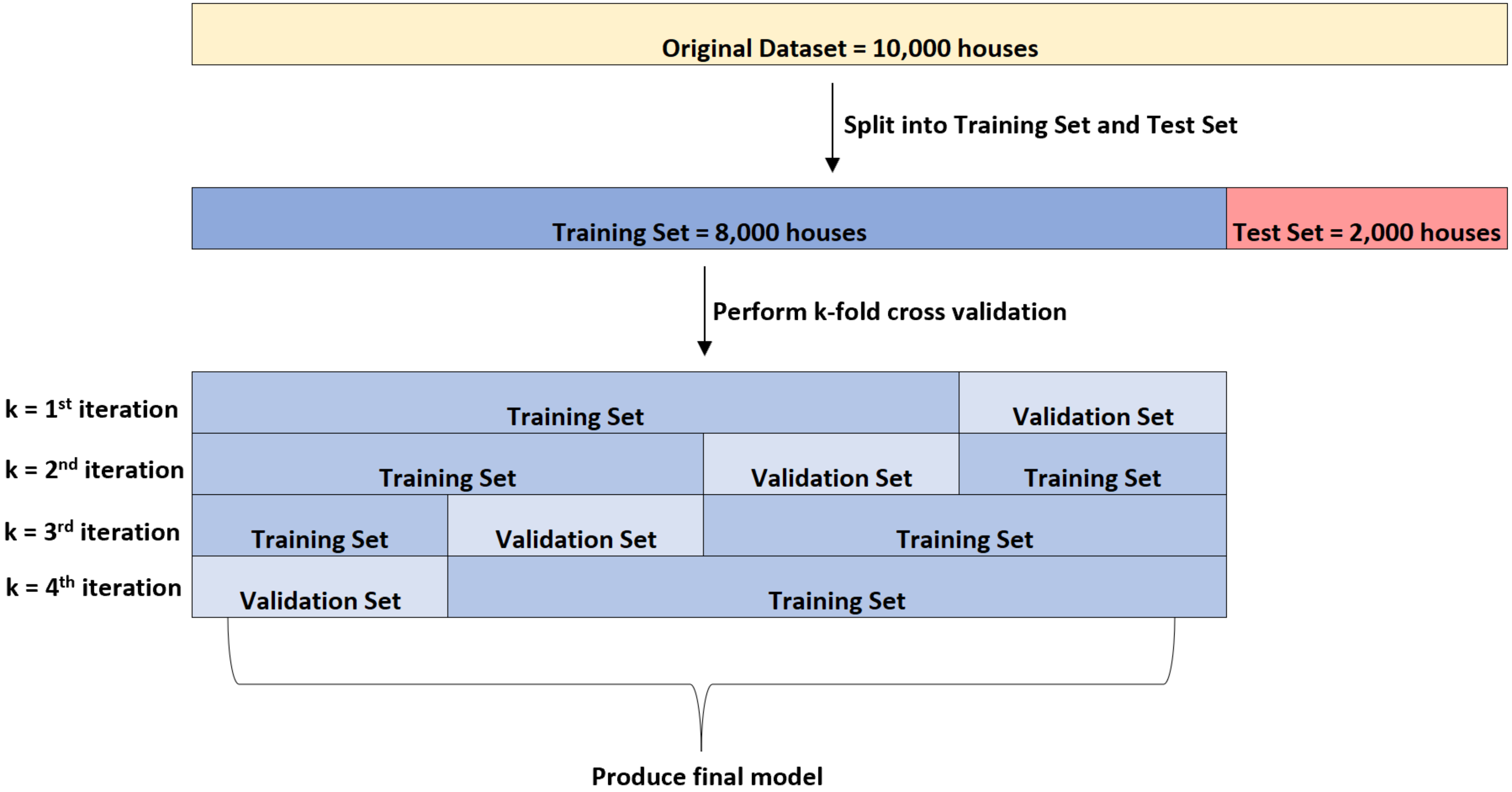
https://colab.research.google.com/drive/1qux4VRnK2Z5A0Q8e-266gt4w7--aqFLQ?usp=sharing

НЕКОТОРЫЕ ОСОБЕННОСТИ МЕДИЦИНСКИХ НАБОРОВ ДАННЫХ



- Если выборка содержит несколько наблюдений от одного пациента (например, параметры QRS-комплексов), то при разделении на подвыборки должно соблюдаться условие – **данные одного пациента не разделяются**!

ПАРА СЛОВ О КРОСС-ВАЛИДАЦИИ



Original Dataset = 10,000 houses

Split into Training Set and Test Set

Training Set = 8,000 houses | Test Set = 2,000 houses

Perform k-fold cross validation

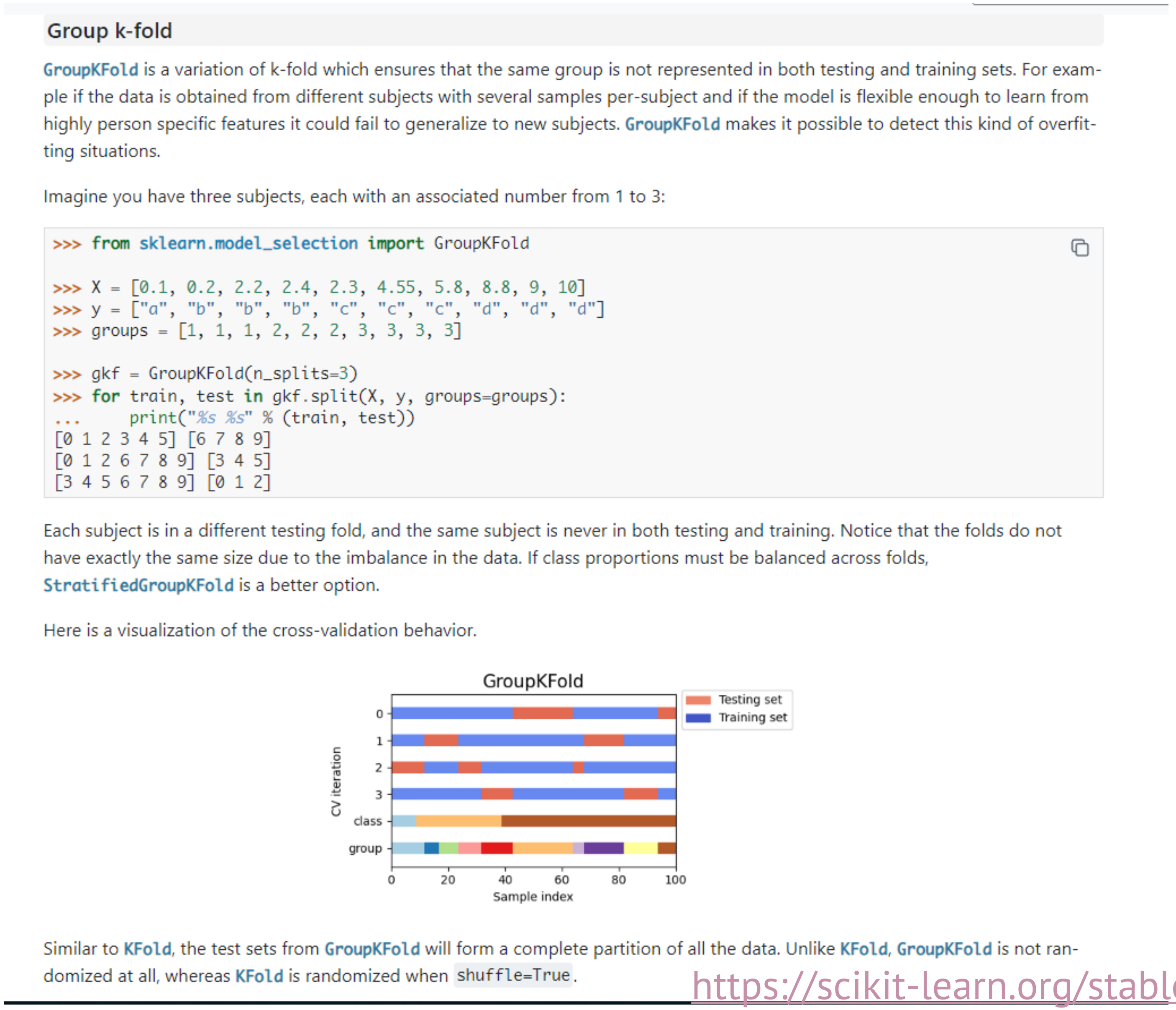
k = 1st iteration: Training Set | Validation Set

k = 2nd iteration: Training Set | Validation Set | Training Set

k = 3rd iteration: Training Set | Validation Set | Training Set

k = 4th iteration: Validation Set | Training Set

Produce final model



Group k-fold

GroupKFold is a variation of k-fold 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 per-subject 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 sklearn.model_selection import GroupKFold

>>> X = [0.1, 0.2, 2.2, 2.4, 2.3, 4.55, 5.8, 8.8, 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 = GroupKFold(n_splits=3)
>>> for train, test in gkf.split(X, y, groups=groups):
...     print("%s %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. If class proportions must be balanced across folds, StratifiedGroupKFold is a better option.

Here is a visualization of the cross-validation behavior.

Similar to KFold, the test sets from GroupKFold will form a complete partition of all the data. Unlike KFold, GroupKFold is not randomized at all, whereas KFold is randomized when shuffle=True.

https://scikit-learn.org/stable/modules/cross_validation.html#group-k-fold