**Cross - Validation and its Types**

* If we split our data into training data and testing data, aren’t we going to lose some important information that the test dataset may hold?
* Cross-validation is a technique for validating the model efficiency by training it on the subset of input data and testing on previously unseen subset of the input data.
* Itis a technique for evaluating ML models by training several ML models on subsets of the available input data and evaluating them on the complementary subset of the data
* Use cross-validation to detect overfitting, ie, failing to generalize a pattern.
* You need some kind of assurance that your model has got most of the patterns from the data correct, and its not picking up too much on the noise, or in other words its low on bias and variance.

The three steps involved in cross-validation are as follows :

1. Reserve some portion of sample data-set.
2. Using the rest data-set train the model.
3. Test the model using the reserve portion of the data-set.

**Types of Cross Validation :** Non-exhaustive and Exhaustive Methods

**Exhaustive Methods**:Test on all possible ways to divide the original sample into a training and a validation set.

1. Leave-P-Out cross validation
2. Leave-one-out cross validation

**Non-exhaustive**: Do not compute all ways of splitting the original data.

1. Holdout Method.
2. K fold cross validation
3. Stratified K fold Cross Validation

**Leave-P-Out cross validation:**

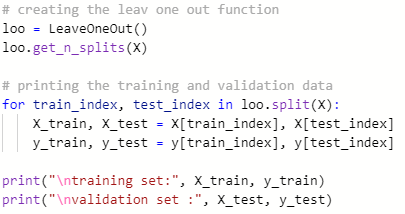
* In this approach, the p datasets are left out of the training data.
* It means, if there are total n datapoints in the original input dataset, then n-p data points will be used as the training dataset and the p data points as the validation set.
* This complete process is repeated for all the samples, and the average error is calculated to know the effectiveness of the model.

There is a disadvantage of this technique; that is, it can be computationally difficult for the large p.

**Leave one out cross-validation**

* This method is similar to the leave-p-out cross-validation, but instead of p, we need to take 1 dataset out of training.
* It means, in this approach, for each learning set, only one datapoint is reserved, and the remaining dataset is used to train the model.
* This process repeats for each datapoint. Hence for n samples, we get n different training set and n test set.
* It has the following features:

1. In this approach, the bias is minimum as all the data points are used.
2. The process is executed for n times; hence execution time is high.
3. This approach leads to high variation in testing the effectiveness of the model as we iteratively check against one data point.



**Holdout Method:**

* This method is the simplest cross-validation technique among all. In this method, we need to remove a subset of the training data and use it to get prediction results by training it on the rest part of the dataset.
* The error that occurs in this process tells how well our model will perform with the unknown dataset. Although this approach is simple to perform, it still faces the issue of high variance, and it also produces misleading results sometimes.

**K fold Cross Validation :**

When we split our training dataset to get a validation set, there’s always a risk of losing some crucial data from the training set, or of losing patterns which might go unnoticed by the model.

This will in turn lead to overfitting or underfitting.

To avoid this we need enough amount of data in both the training set and the validation set. And for this, we use K-Fold Cross Validation.

In this method, the original training set is divided into k subsets.

The holdout method is now repeated k times with different datasets.

In each fold, one of the k subsets is taken as the validation set, and the remaining k – 1 subsets are used as the training set.

The error estimations from all the folds are taken and averaged to give us the final error estimation of the model.

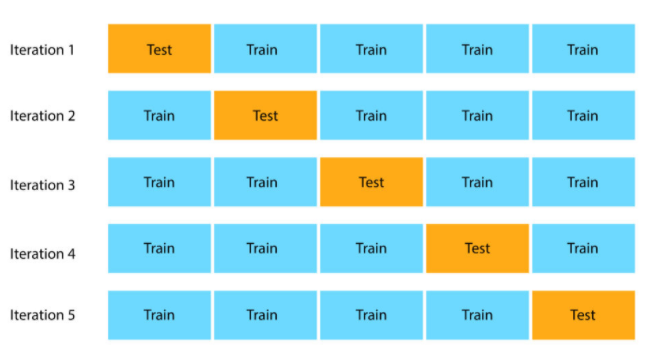
Because we’re using all the k sets for validation, each datapoint appears in the validation set exactly once. And each point of data appears in the training set exactly k – 1 times.

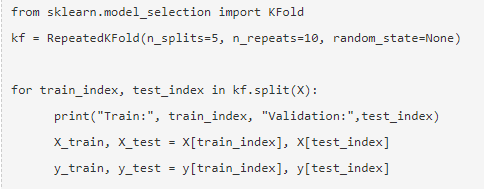
This greatly improves the accuracy of the model.

This also reduces bias as most of the data is being used for fitting, and reduces variance as most of the data is also being used for validation.

And because we’re interchanging the data in each fold, it improves the overall efficiency of the model.

Depending on your dataset, you can select a k value of your own. But in most cases, k = 5 or k = 10 is preferred.





**Stratified K-Fold Cross Validation:**

* Stratified k-fold cross-validation is same as just k-fold cross-validation, But in Stratified k-fold crossvalidation, it does stratified sampling instead of random sampling.
* One obvious problem with normal KFold, is that each in each fold the distribution of classes in the validation set, will be not be same.
* This is a big problem with imbalanced datasets.
* To overcome this problem we will use Stratified-KFold Validation. StratifiedKFold ensures that each of the splits have same proportion of examples of each class.
* StratifiedKFold is a variation of KFold. First, StratifiedKFold shuffles your data, after that splits the data into n\_splits parts and Done.
* Now, it will use each part as a test set. Note that it only and always shuffles data one time before splitting.

