# Model Selection Notes

**Model selection** is a technique for selecting the best model after the individual models are evaluated based on the required criteria.

## *Resampling methods*

Resampling methods, as the name suggests, are simple techniques of rearranging data samples to inspect if the model performs well on data samples that it has not been trained on. In other words, resampling helps us understand **if the model will generalize well**.

## *Random Split*

Random Splits are used to randomly sample a percentage of data into training, testing, and preferably validation sets. The advantage of this method is that there is a good chance that the original population is well represented in all the three sets. In more formal terms, random splitting will prevent a biased sampling of data.

It is very important to note the use of the validation set in model selection. The validation set is the second test set and one might ask, why have two test sets?

In the process of feature selection and model tuning, the test set is used for model evaluation. This means that the model parameters and the feature set are selected such that they give an optimal result on the test set. Thus, the validation set which has completely unseen data points (not been used in the tuning and feature selection modules) is used for the final evaluation.

## *Time-Based Split*

There are **some types of data where random splits are not possible**. For example, if we have to train a model for weather forecasting, we cannot randomly divide the data into training and testing sets. This will jumble up the seasonal pattern! Such data is often referred to by the term – Time Series.

In such cases, a time-wise split is used. The training set can have data for the last three years and 10 months of the present year. The last two months can be reserved for the testing or validation set.

There is also a **concept of *window* sets** – where the model is trained till a particular date and tested on the future dates iteratively such that the training window keeps increasing shifting by one day (consequently, the test set also reduces by a day). The advantage of this method is that it stabilizes the model and prevents overfitting when the test set is very small (say, 3 to 7 days).

However, the drawback of time-series data is that the events or data points are not ***mutually independent***. One event might affect every data input that follows after.

For instance, a change in the governing party might considerably change the population statistics for the years to follow. Or the infamous coronavirus pandemic is going to have a massive impact on economic data for the next few years.

No machine learning model can learn from past data in such a case because the data points before and after the event have major differences.

## *K-Fold Cross-Validation*

Cross-validation is a statistical method used to estimate the skill of machine learning models.

It is commonly used in applied machine learning to compare and select a model for a given predictive modeling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods.

In this tutorial, you will discover a gentle introduction to the k-fold cross-validation procedure for estimating the skill of machine learning models.

## k-Fold Cross-Validation

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.

The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

The general procedure is as follows:

1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. For each unique group:
    1. Take the group as a hold out or test data set
    2. Take the remaining groups as a training data set
    3. Fit a model on the training set and evaluate it on the test set
    4. Retain the evaluation score and discard the model
4. Summarize the skill of the model using the sample of model evaluation scores

Importantly, each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure. This means that each sample is given the opportunity to be used in the hold out set 1 time and used to train the model k-1 times.

*This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining k − 1 folds.*

It is also important that any preparation of the data prior to fitting the model occur on the CV-assigned training dataset within the loop rather than on the broader data set. This also applies to any tuning of hyperparameters. A failure to perform these operations within the loop may result in data leakage and an optimistic estimate of the model skill.

## What is meant by Data Leakage?

**Data Leakage** is the scenario where the Machine Learning Model is already aware of some part of test data after training.This causes the problem of overfitting.

In Machine learning, **Data Leakage** refers to a mistake that is made by the creator of a machine learning model in which they accidentally share the information between the test and training data sets. Typically, when splitting a data set into testing and training sets, the goal is to ensure that no data is shared between these two sets. Ideally, there is no intersection between these two sets. This is because the purpose of the testing set is to simulate the real-world data which is unseen to that model. However, when evaluating a model, we do have full access to both our train and test sets, so it is our duty to ensure that there is no overlapping between the training data and the testing data (i.e, no intersection).

As a result, due to the Data leakage, we got unrealistically high levels of performance of our model on the test set, because that model is being run on data that it had already seen in some capacity in the training set. The model effectively memorizes the training set data and is easily able to correctly output the labels or values for those examples of the test dataset. Clearly, this is not ideal, as it misleads the person who evaluates the model. When such a model is then used on truly unseen data that is coming mostly on the production side, then the performance of that model will be much lower than expected after deployment.

## ***Stratified K-Fold***

The process for stratified K-Fold is similar to that of K-Fold cross-validation with one single point of difference – **unlike in k-fold cross-validation, the values of the target variable is taken into consideration in stratified k-fold.**

If for instance, the target variable is a categorical variable with 2 classes, then stratified k-fold ensures that each test fold gets an equal ratio of the two classes when compared to the training set.

This makes the model evaluation more accurate and the model training less biased.

**Pros:**

1. **Works perfectly well for Imbalanced Data:** Each fold in stratified cross-validation will have a representation of data of all classes in the same ratio as in the whole dataset.

**Cons:**

**1. Not suitable for Time Series data:** For Time Series data the order of the samples matter. But in Stratified Cross-Validation, samples are selected in random order.

***Bootstrap***

Bootstrap is one of the most powerful ways to obtain a stabilized model. It is close to the random splitting technique since it follows the concept of random sampling.

The first step is to select a sample size (which is usually equal to the size of the original dataset). Thereafter, a sample data point must be randomly selected from the original dataset and added to the bootstrap sample. After the addition, the sample needs to be put back into the original sample. This process needs to be repeated for N times, where N is the sample size.

Therefore, it is a resampling technique that creates the bootstrap sample by sampling data points from the original dataset ***with replacement***. This means that the bootstrap sample can contain multiple instances of the same data point.

The model is trained on the bootstrap sample and then evaluated on all those data points that did not make it to the bootstrapped sample. These are called the ***out-of-bag*** samples.

Bootstrap Break down

In statistics, Bootstrap Sampling is a method that involves drawing of sample data repeatedly with replacement from a data source to estimate a population parameter.

- **Sampling:** With respect to statistics, sampling is the process of selecting a subset of items from a vast collection of items (population) to estimate a certain characteristic of the entire population
- **Sampling with replacement:** It means a data point in a drawn sample can reappear in future drawn samples as well
- **Parameter estimation:** It is a method of estimating parameters for the population using samples. A parameter is a measurable characteristic associated with a population. For example, the average height of residents in a city, the count of red blood cells, etc.

## Probabilistic measures

Probabilistic Measures do **not just take into account the model performance but also the model complexity**. Model complexity is the measure of the model's ability to capture the variance in the data.

For example, a highly biased model like the linear regression algorithm is less complex and on the other hand, a neural network is very high on complexity.

Another important point to note here is that the **model performance** taken into account in probabilistic measures is **calculated from the training set only**. A hold-out test set is typically not required.

A fair bit of disadvantage however lies in the fact that probabilistic measures do not consider the uncertainty of the models and has a chance of selecting simpler models over complex models.

### ***Akaike Information Criterion (AIC)***

It is common knowledge that every model is not completely accurate. There is always some information loss which can be measured using the KL information metric. Kulback-Liebler or KL divergence is the measure of the difference in the probability distribution of two variables.

A statistician, Hirotugu Akaike, took into consideration the relationship between KL Information and Maximum Likelihood (in maximum-likelihood, one wishes to maximize the conditional probability of observing a datapoint X, given the parameters and a specified probability distribution) and developed the concept of Information Criterion (or IC). Therefore, Akaike's IC or AIC is the measure of information loss. This is how the discrepancy between two different models is captured and the model with the least information loss is suggested as the model of choice.

$$AIC = (2K - 2log(L))/N$$

- K = number of independent variables or predictors
- L = maximum-likelihood of the model
- N = number of data points in the training set (especially helpful in case of small datasets)

The limitation of AIC is that it is not very good with generalizing models as it tends to select complex models that lose less training information.

## ***Bayesian Information Criterion (BIC)***

BIC was derived from the Bayesian probability concept and is suited for models that are trained under the maximum likelihood estimation.

$$BIC = K * log(N) - 2log(L)$$

- K = number of independent variables
- L = maximum-likelihood
- N = Number of sampler/data points in the training set

BIC penalizes the model for its complexity and is preferably used when the size of the dataset is not very small (otherwise it tends to settle on very simple models).

## ***Minimum Description Length (MDL)***

MDL is derived from the Information theory which deals with quantities such as entropy that measure the average number of bits required to represent an event from a probability distribution or a random variable.

MDL or the minimum description length is the minimum number of such bits required to represent the model.

$$MDL = L(h) + L(D|h)$$

- d = model
- D = predictions made by the model
- L(h) = number of bits required to represent the model
- L(D | h) = number of bits required to represent the predictions from the model

## ***Structural Risk Minimization (SRM)***

Machine learning models face the inevitable problem of defining a generalized theory from a set of finite data. This leads to cases of overfitting where the model gets biased to the training data which is its primary learning source. SRM tries to balance out the model's complexity against its success at fitting on the data.