# WEEK 6

**Why do we need data splitting?**

Whenever we train a machine learning model, we can't train that model on a single dataset or even we train it on a single dataset then we will not be able to assess the performance of our model. For that reason, we split our source data into training, testing, and validation datasets. Now for understanding the need for data split let's take an example of classroom teaching.

Suppose a mathematics faculty teaches her students about an algorithm. For the explanation the teacher uses some examples, those examples are our training dataset. The student in this case is our machine learning model and the examples are part of the dataset. Because students are learning by those examples that's why we call it our training set.

And to check whether the students got the concepts of the algorithm correctly, the teacher give some practice problems to the students. By solving those problems students will evaluate their learning and if there is any difficulty they face, they will ask their doubt of the instructor (Feedback of model).

There might be some misunderstanding between the students for some concepts because of which they were not able to solve the problem. So, the teacher might try to explain the problem to the students in a different way (Fine-tuning of parameters).

Students can also improve their learning by solving more and more practice problems (Validation). The more diverse the practice problems are great will be the learning (Cross-validation). Students can improve their accuracy (Cross-validation accuracy) by repeatedly solving practice problems.

But once the class teaching is over and the exam comes, there is no going back to the teacher or solving practice problems. Whatever the students have learned, they need to use it for solving the problems given in the exam (Testing data). And the result that the student gets will be the final accuracy about how well that student learned about that concept.

The summary of this analogy is:
Training data = Classroom teaching
Validation data = Practice problems
Testing data = Exam questions

## Data Splitting

- The train-test split is a technique for evaluating the performance of a machine learning algorithm.
- It can be used for classification or regression problems and can be used for any supervised learning algorithm.
- The procedure involves taking a dataset and dividing it into two subsets.
- The first subset is used to fit the model and is referred to as the training dataset.
- The second subset is not used to train the model; instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values.
- This second dataset is referred to as the test dataset.

## Components to split

When we decide to split the data, we should know how many splits of data we need. Generally, there are three partitions of data we make, which are:

1. **The Training Set**: It is the set of data that is used to train and make the model learn the hidden features/patterns in the data.
2. **The Validation Set**: The validation set is a set of data, separate from the training set, that is used to validate our model performance **during training**.
3. **The Test Set**: The test set is a separate set of data used to test the model **after completing** the training.


### Overfitting and Underfitting in Machine Learning

- Overfitting and Underfitting are the two main problems that occur in machine learning and degrade the performance of the machine learning models.
- The main goal of each machine learning model is **to generalize well**.
- Here **generalization** defines the ability of an ML model to provide a suitable output by adapting the given set of unknown input.
- It means after providing training on the dataset, it can produce reliable and accurate output.
- Hence, the underfitting and overfitting are the two terms that need to be checked for the performance of the model and whether the model is generalizing well or not.

### Bias

In general, a machine learning model analyses the data, find patterns in it and make predictions. While training, the model learns these patterns in the dataset and applies them to test data for prediction. ***While making predictions, a difference occurs between prediction values made by the model and actual values/expected values, and this difference is known as bias errors or Errors due to bias***.

- **Low Bias:** A low bias model will make fewer assumptions about the form of the target function.
- **High Bias:** A model with a high bias makes more assumptions, and the model becomes unable to capture the important features of our dataset.

### Variance:

If the machine learning model performs well with the training dataset, but does not perform well with the test dataset, then variance occurs.

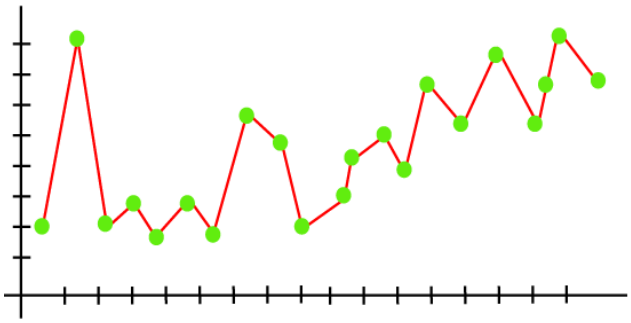
## Overfitting

Overfitting occurs when our machine learning model tries to cover all the data points or more than the required data points present in the given dataset. Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy of the model. The overfitted model has **low bias** and **high variance.**

The chances of occurrence of overfitting increase as much we provide training to our model. It means the more we train our model, the more chances of occurring the overfitted model.

Overfitting is the main problem that occurs in supervised learning

**Example:** The concept of the overfitting can be understood by the below graph of the linear regression output:



As we can see from the above graph, the model tries to cover all the data points present in the scatter plot. It may look efficient, but in reality, it is not so. Because the goal of the regression model to find the best fit line, but here we have not got any best fit, so, it will generate the prediction errors.

### How to avoid the Overfitting in Model

Both overfitting and underfitting cause the degraded performance of the machine learning model. But the main cause is overfitting, so there are some ways by which we can reduce the occurrence of overfitting in our model.

- **Cross-Validation**
- **Training with more data**
- **Removing features**
- **Early stopping the training**
- **Regularization**

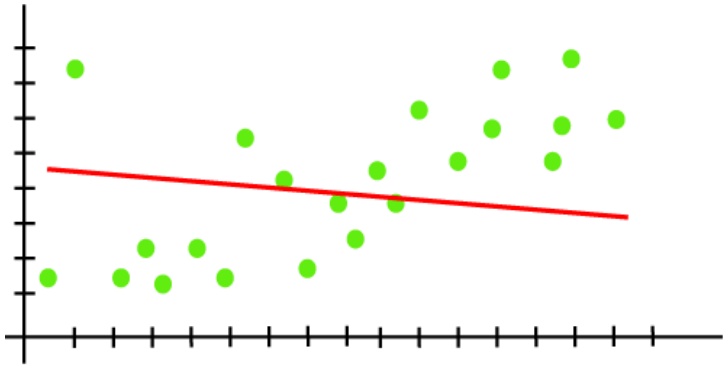
## Underfitting

Underfitting occurs when our machine learning model is not able to capture the underlying trend of the data. To avoid the overfitting in the model, the fed of training data can be stopped at an early stage, due to which the model may not learn enough from the training data. As a result, it may fail to find the best fit of the dominant trend in the data.

In the case of underfitting, the model is not able to learn enough from the training data, and hence it reduces the accuracy and produces unreliable predictions.

An underfitted model has high bias and low variance.

**Example:** We can understand the underfitting using below output of the linear regression model:



As we can see from the above diagram, the model is unable to capture the data points present in the plot.

**How to avoid underfitting:**

- By increasing the training time of the model.
- By increasing the number of features.

**Split training and testing data sets in Python using train_test_split() of sci-kit learn**

```
from sklearn.model_selection import train_test_split
```

```
y = df['sales']
X = df.drop('sales',axis=1)
```

```
# splitting the dataframe into train and test sets
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=101)
```

## What is Regression?

- Regression is defined as a statistical method that helps us to analyze and understand the relationship between two or more variables of interest.
- In regression, we normally have one dependent variable and one or more independent variables.
- Here we try to guess the value of the dependent variable “Y” with the help of the independent variables.
- In other words, we are trying to understand, how the value of ‘Y’ changes w.r.t change in ‘X’.

In regression analysis we need to understand the following terms:

- **Dependent Variable:** This is the variable that we are trying to understand or forecast.
- **Independent Variable:** These are factors that influence the analysis or target variable and provide us with information regarding the relationship of the variables with the target variable.

## Types of Regression

### 1. Linear regression

One of the most basic types of regression in machine learning, linear regression comprises a predictor variable and a dependent variable related to each other in a linear fashion. Linear regression involves the use of a best fit line. You should use linear regression when your variables are related linearly. For example, if you are forecasting the effect of increased advertising spend on sales. However, this analysis is susceptible to outliers, so it should not be used to analyze big data sets.

### 2. Logistic regression

If the dependent variable have a discrete value, logistic regression is used. In other words, if the dependent variable can take only one of two values (either 0 or 1, true or false, black or white, spam or not spam, and so on), we use logistic regression to analyze the data.

Logistic regression uses a sigmoid curve to show the relationship between the target and independent variables. However, logistic regression works best with large data sets that have an almost equal occurrence of values in target variables. The dataset should not contain a high correlation between independent variables (a phenomenon known as multicollinearity), as this will create a problem when ranking the variables.

### 3. Ridge regression

If there is a high correlation between independent variables, ridge regression is a more suitable tool. It is known as a regularization technique, and is used to reduce the complexity of the model. It introduces a small amount of bias (known as the 'ridge regression penalty') which makes the model less susceptible to overfitting.

### 4. Lasso regression

Like ridge regression, lasso regression is another regularization technique that reduces the model's complexity. It can use feature selection, letting you select a set of features from the dataset to build the model. By only using the required features – and setting the rest as zero – lasso regression avoids overfitting.

### 5. Polynomial regression

Polynomial regression models a non-linear dataset using a linear model. It is the equivalent of making a square peg fit into a round hole. It works in a similar way to multiple linear regression, but uses a non-linear curve. It is used when data points are present in a non-linear fashion.

The model transforms these data points into polynomial features of a given degree, and models them using a linear model. This involves best fitting them using a polynomial line, which is curved, rather than the straight line seen in linear regression.

## Regularization in Machine Learning

- Regularization is one of the most important concepts of machine learning. It is a technique to prevent the model from overfitting by adding extra information to it.

- Sometimes the machine learning model performs well with the training data but does not perform well with the test data.
- It means the model is not able to predict the output when deals with unseen data by introducing noise in the output, and hence the model is called overfitted. This problem can be deal with the help of a regularization technique.
- This technique can be used in such a way that it will allow to maintain all variables or features in the model by reducing the magnitude of the variables. Hence, it maintains accuracy as well as a generalization of the model.
- It mainly regularizes or reduces the coefficient of features toward zero. In simple words, ***"In regularization technique, we reduce the magnitude of the features by keeping the same number of features."***

There are mainly two types of regularization techniques, which are given below:

- **Ridge Regression**
- **Lasso Regression**

## Real-world examples of linear regression models

The following represents some real-world examples / use cases where linear regression models can be used:

- **Forecasting sales**: Organizations often use linear regression models to forecast future sales. This can be helpful for things like budgeting and planning. Algorithms such as Amazon's item-to-item collaborative filtering are used to predict what customers will buy in the future based on their past purchase history.
- **Cash forecasting**: Many businesses use linear regression to forecast how much cash they'll have on hand in the future. This is important for things like managing expenses and ensuring that there is enough cash on hand to cover unexpected costs.
- **Analyzing survey data**: Linear regression can also be used to analyze survey data. This can help businesses understand things like customer satisfaction and product preferences. For example, a company might use linear regression to figure out how likely people are to recommend their product to others.
- **Stock predictions**: A lot of businesses use linear regression models to predict how stocks will perform in the future. This is done by analyzing past data on stock prices and trends to identify patterns.
- **Predicting consumer behavior**: Businesses can use linear regression to predict things like how much a customer is likely to spend. Regression models can also be used to predict consumer behavior. This can be helpful for things like targeted marketing and product development. For example, Walmart uses linear regression to predict what products will be popular in different regions of the country.
- **Analysis of relationship between variables**: Linear regression can also be used to identify relationships between different variables. For example, you could use linear regression to find out how temperature affects ice cream sales.

## Linear Regression

- Linear regression is an algorithm that provides a **linear relationship between an independent variable and a dependent variable** to predict the outcome of future events.
- It is a statistical method used in data science and machine learning for predictive analysis.
- The independent variable is also the predictor or explanatory variable that remains unchanged due to the change in other variables.
- However, the dependent variable changes with fluctuations in the independent variable.
- The regression model predicts the value of the dependent variable, which is the response or outcome variable being analysed or studied.
- Thus, linear regression is **a supervised learning algorithm** that simulates a mathematical relationship between variables and makes predictions for continuous or numeric variables such as sales, salary, age, product price, etc.
- A sloped straight line represents the linear regression model.


**Best Fit Line for a Linear Regression Model**

In the above figure,

X-axis = Independent variable

Y-axis = Output / dependent variable

Line of regression = Best fit line for a model

- Here, a line is plotted for the given data points that suitably fit all the issues.
- Hence, it is called the 'best fit line.'
- The goal of the linear regression algorithm is to find this best fit line.

### Linear Regression Equation

- The regression model defines a linear function between the X and Y variables that best showcases the relationship between the two.

- It is represented by the slant line seen in the above figure, where the objective is to determine an optimal 'regression line' that best fits all the individual data points.

Mathematically these slant lines follow the following equation,

$$Y = m*X + b$$

Where X = dependent variable (target)

Y = independent variable

m = slope of the line

b = y-intercept

**Types of Linear Regression**

1. **Simple Linear Regression**: Regression involving only one independent variable is referred as simple linear regression.

2. **Multiple Linear Regression:** A regression model for involving multiple independent variables is called as multiple linear regression.

   The equation for multiple linear regression is similar to the equation for a simple linear equation, i.e., y(x) = p0 + p1x1 plus the additional weights and inputs for the different features which are represented by p(n)x(n). The formula for multiple linear regression would look like,

   $$y(x) = p0 + p1x1 + p2x2 + \ldots + p(n)x(n)$$

   The machine learning model uses the above formula and different weight values to draw lines to fit. Moreover, to determine the line best fits the data, the model evaluates different weight combinations that best fit the data and establishes a strong relationship between the variables.

3. **Polynomial Regression:** Polynomial regression is a form of Linear regression where only due to the Non-linear relationship between dependent and independent variables we add some polynomial terms to linear regression to convert it into Polynomial regression.

   Suppose we have X as Independent data and Y as dependent data. Before feeding data to a mode in preprocessing stage we convert the input variables into polynomial terms using some degree.

   The equation of polynomial becomes something like this.

   $$\mathbf{y = a_0 + a_1x_1 + a_2x_1^2 + \ldots + a_nx_1^n}$$

   The degree of order which to use is a Hyperparameter, and we need to choose it wisely. But using a high degree of polynomial tries to overfit the data and for smaller values of degree, the model tries to underfit so we need to find the optimum value of a degree.

## The Four Assumptions of Linear Regression

Linear regression is a useful statistical method we can use to understand the relationship between two variables, x and y. However, before we conduct linear regression, we must first make sure that four assumptions are met:

1. **Linear relationship**: There exists a linear relationship between the independent variable, x, and the dependent variable, y.

2. **Independence**: The residuals are independent. In particular, there is no correlation between consecutive residuals in time series data.

3. **Homoscedasticity**: The residuals have constant variance at every level of x.

4. **Normality**: The residuals of the model are normally distributed.

If one or more of these assumptions are violated, then the results of our linear regression may be unreliable or even misleading.

## Assumptions in Multiple Linear Regression:

Before we perform multiple linear regression, we must first make sure that five assumptions are met:

**1. Linear relationship:** There exists a linear relationship between each predictor variable and the response variable.

**2. No Multicollinearity:** None of the predictor variables are highly correlated with each other.

**3. Independence:** The observations are independent.

**4. Homoscedasticity:** The residuals have constant variance at every point in the linear model.

**5. Multivariate Normality:** The residuals of the model are normally distributed.

## Gradient Descent

- Linear regression is about finding the line of best fit for a dataset. This line can then be used to make predictions.
- **Gradient descent is a tool to arrive at the line of best fit**
- In gradient descent, you start with a random line. Then you change the parameters of the line (i.e. slope and y-intercept) little by little to arrive at the line of best fit.
- How do you know when you arrived at the line of best fit?
- For every line you try — line A, line B, line C, etc — you calculate the sum of squares of the errors. If line B has a smaller value than line A, then line B is a better fit, etc.
- **Gradient descent is an algorithm that approaches the least squared regression line via minimizing sum of squared errors through multiple iterations.**

**At a high level, this is how gradient descent works:**

- You start with a random line, let's say line A. You compute the sum of squared errors for that line.
- Then, you adjust your slope and y-intercept.
- You compute the sum of squared errors again for your new line.
- You continue adjusting until you reach a local minimum, where the sum of squared errors is the smallest and additional tweaks does not produce better result.
- The way you adjust your slope and intercept will be covered in more details momentarily.

# Model Evaluation & testing

## Evaluate regression model:

- Model evaluation is very important in data science.
- It helps you to understand the performance of your model and makes it easy to present your model to other people.
- There are many different evaluation metrics out there but only some of them are suitable to be used for regression.
- In regression, it is impossible for you to predict the exact value but rather **how close your prediction is against the real value**.

## Main metrics for model evaluation in regression:

### 1. R Square/Adjusted R Square

R Square measures how much variability in dependent variable can be explained by the model. It is the square of the Correlation Coefficient(R) and that is why it is called R Square.

$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

- R Square is calculated by the sum of squared of prediction error divided by the total sum of the square which replaces the calculated prediction with mean.
- R Square value is between 0 to 1 and a bigger value indicates a better fit between prediction and actual value.
- R Square is a good measure to determine how well the model fits the dependent variables.
- **However, it does not take into consideration of overfitting problem**.
- If your regression model has many independent variables, because the model is too complicated, it may fit very well to the training data but performs badly for testing data.
- That is why Adjusted R Square is introduced because it will penalize additional independent variables added to the model and adjust the metric to prevent overfitting issues.

### 2. Mean Square Error(MSE)/Root Mean Square Error(RMSE)

While R Square is a relative measure of how well the model fits dependent variables, Mean Square Error is an absolute measure of the goodness for the fit.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

- MSE is calculated by the sum of square of prediction error which is real output minus predicted output and then divide by the number of data points.
- It gives you an absolute number on how much your predicted results deviate from the actual number.

- You cannot interpret many insights from one single result but it gives you a real number to compare against other model results and help you select the best regression model.
- Root Mean Square Error(RMSE) is the square root of MSE.
- It is used more commonly than MSE because firstly sometimes MSE value can be too big to compare easily.
- Secondly, MSE is calculated by the square of error, and thus square root brings it back to the same level of prediction error and makes it easier for interpretation.

## Optimize Linear Regression models

- Suppose the data-set available contains hundreds of different features and the corresponding target values to train our regression model and get estimates of coefficients(b0, b1,b2….).
- However, with limited computation power the conundrum of not only choosing the appropriate features but also estimating the optimal number of features to be used, prevails.
- Thanks to the built in python libraries like scikit learn and numpy which readily bring forth estimates for regression coefficients. N
- ow we would dive into the methodology of choosing the appropriate features and the number of features we want to be present in our regression equation.
- Let our variable to be predicted be called — 'Y' .
- Our primary objective is to get the correlation coefficients of Y and each of our independent variables separately.
- We can draw a heatmap of correlation matrix of all the variables in the dataset.
- Prepare and pre-process the features having highest correlation coefficient with the dependent variable and select them to be a part of the regression equation.
- If two or more independent variables have a high degree of correlation, then it is better to include any one of those features.
- Too many independent variables will over-fit the training data and result in a not so good regression model.

## What is Cross Validation?

- Cross-validation is a statistical method used to estimate the performance (or accuracy) of machine learning models.
- It allows us to utilize our data better.
- It is used to protect against overfitting in a predictive model, particularly in a case where the amount of data may be limited.
- In cross-validation, you make a fixed number of folds (or partitions) of the data, run the analysis on each fold, and then average the overall error estimate.

- When dealing with a Machine Learning task, you have to properly identify the problem so that you can pick the most suitable algorithm which can give you the best score. But how do we compare the models?
- Say, you have trained the model with the dataset available and now you want to know how well the model can perform.
- One approach can be that you are going to test the model on the dataset you have trained it on, but this may not be a good practice.
- So, what is wrong with testing the model on the training dataset?
- If we do so, we assume that the training data represents all the possible scenarios of real-world and this will surely never be the case.
- Our main objective is that the model should be able to work well on the real-world data, although the training dataset is also real-world data, it represents a small set of all the possible data points(examples) out there.
- So, to know the real score of the model, it should be tested on the data that it has never seen before and this set of data is usually called testing set.
- But if we split our data into training data and testing data, we are going to lose some important information that the test dataset may hold

**Types of Cross Validation**

There are different types of cross validation methods, and they could be classified into two broad categories – Non-exhaustive and Exhaustive Methods. We're going to look at a few examples from both the categories.

**1. Non-exhaustive Methods**

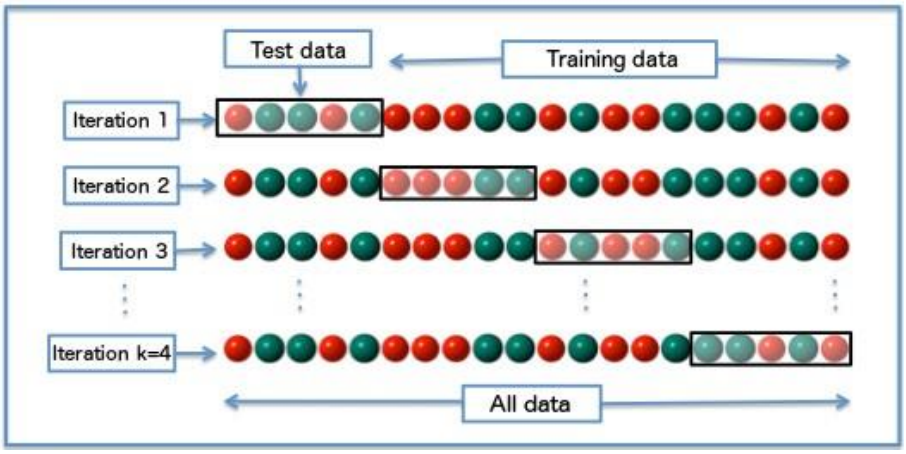
Non-exhaustive cross validation methods, as the name suggests do not compute all ways of splitting the original data. Let us go through the methods to get a clearer understanding.

**a. Holdout method**

- This is a quite basic and simple approach in which we divide our entire dataset into two parts viz- training data and testing data.
- As the name, we train the model on training data and then evaluate on the testing set.
- Usually, the size of training data is set more than twice that of testing data, so the data is split in the ratio of 70:30 or 80:20.
- In this approach, the data is first shuffled randomly before splitting.
- As the model is trained on a different combination of data points, the model can give different results every time we train it, and this can be a cause of instability.
- Also, we can never assure that the train set we picked is representative of the whole dataset.
- Also when our dataset is not too large, there is a high possibility that the testing data may contain some important information that we lose as we do not train the model on the testing set.
- The hold-out method is good to use when you have a very large dataset, you're on a time crunch, or you are starting to build an initial model in your data science project.

**b. K fold cross validation**

- K-fold cross validation is one way to improve the holdout method.
- This method guarantees that the score of our model does not depend on the way we picked the train and test set.
- The data set is divided into k number of subsets and the holdout method is repeated k number of times. Let us go through this in steps:
    1. Randomly split your entire dataset into k number of folds (subsets)
    2. For each fold in your dataset, build your model on k – 1 folds of the dataset. Then, test the model to check the effectiveness for kth fold
    3. Repeat this until each of the k-folds has served as the test set
    4. The average of your k recorded accuracy is called the cross-validation accuracy and will serve as your performance metric for the model.




- Because it ensures that every observation from the original dataset has the chance of appearing in training and test set, this method generally results in a less biased model compare to other methods.
- It is one of the best approaches if we have limited input data.
- The disadvantage of this method is that the training algorithm has to be rerun from scratch k times, which means it takes k times as much computation to make an evaluation.

**c. Stratified K Fold Cross Validation**

- Using K Fold on a classification problem can be tricky.
- Since we are randomly shuffling the data and then dividing it into folds, chances are we may get highly imbalanced folds which may cause our training to be biased.
- For example, let us somehow get a fold that has majority belonging to one class(say positive) and only a few as negative class.
- This will certainly ruin our training and to avoid this we make stratified folds using stratification.
- Stratification is the process of rearranging the data so as to ensure that each fold is a good representative of the whole.
- For example, in a binary classification problem where each class comprises of 50% of the data, it is best to arrange the data such that in every fold, each class comprises of about half the instances.

**2. Exhaustive Methods**

Exhaustive cross validation methods and test on all possible ways to divide the original sample into a training and a validation set.

**a. Leave-P-Out cross validation**

- When using this exhaustive method, we take p number of points out from the total number of data points in the dataset(say n).
- While training the model we train it on these (n – p) data points and test the model on p data points.
- We repeat this process for all the possible combinations of p from the original dataset. Then to get the final accuracy, we average the accuracies from all these iterations.
- This is an exhaustive method as we train the model on every possible combination of data points.
- Remember if we choose a higher value for p, then the number of combinations will be more and we can say the method gets a lot more exhaustive.

**b. Leave-one-out cross validation**

- This is a simple variation of Leave-P-Out cross validation and the value of p is set as one.
- This makes the method much less exhaustive as now for n data points and p = 1, we have n number of combinations.

## Why you should use Cross-Validation?

**1. Use All Your Data:** When we have very little data, splitting it into training and test set might leave us with a very small test set. If we use cross-validation in this case, we build K different models, so we are able to make predictions on **all** of our data.

**2. Get More Metrics:** when we create five different models using our learning algorithm and test it on five different test sets, we can be more confident in our algorithm performance. When we do a single evaluation on our test set, we get only one result. This result may be because of chance or a biased test set for some reason. By training five (or ten) different models we can understand better what's going on.

**3. Work with Dependent/Grouped Data:** When we perform a random train-test split of our data, we assume that our examples are independent. That means that by knowing/seeing some instance will not help us understand other instances. However, that's not always the case. In such cases **We can use cross-validation on the group level.**

**4. Parameters Fine-Tuning:** Most of the learning algorithms require some parameters tuning. We want to find the best parameters for our problem. We do it by trying different values and choosing the best ones. However, we can't do it on our training test and not on our test set of course. We have to use a third set, a validation set. By splitting our data into three sets instead of two, we'll tackle all the such issues. But if we don't have a lot of data, by doing cross-validation, we're able to do all those steps using a single set.