

# MACHINE LEARNING-INTRODUCTION

## What is Machine Learning

In the real world, we are surrounded by humans who can learn everything from their experiences with their learning capability, and we have computers or machines which work on our instructions. But can a machine also learn from experiences or past data like a human does? So here comes the role of **Machine Learning**

## Introduction to Machine Learning

A subset of artificial intelligence known as machine learning focuses primarily on the creation of algorithms that enable a computer to independently learn from data and previous experiences. Arthur Samuel first used the term "machine learning" in 1959. It could be summarized as follows:

Without being explicitly programmed, machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things.

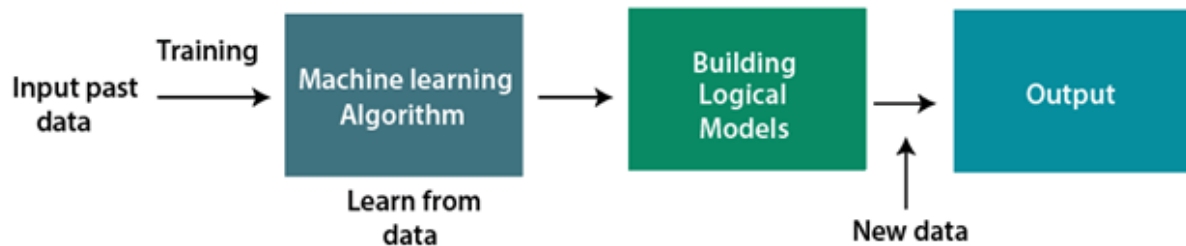
Machine learning algorithms create a mathematical model that, without being explicitly programmed, aids in making predictions or decisions with the assistance of sample historical data, or training data. For the purpose of developing predictive models, machine learning brings together statistics and computer science. Algorithms that learn from historical data are either constructed or utilized in machine learning. The performance will rise in proportion to the quantity of information we provide.

**A machine can learn if it can gain more data to improve its performance.**

## How does Machine Learning work

A machine learning system builds prediction models, learns from previous data, and predicts the output of new data whenever it receives it. The amount of data helps to build a better model that accurately predicts the output, which in turn affects the accuracy of the predicted output.

Let's say we have a complex problem in which we need to make predictions. Instead of writing code, we just need to feed the data to generic algorithms, which build the logic based on the data and predict the output. Our perspective on the issue has changed as a result of machine learning. The Machine Learning algorithm's operation is depicted in the following block diagram:



## Features of Machine Learning:

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

## Need for Machine Learning

The demand for machine learning is steadily rising. Because it is able to perform tasks that are too complex for a person to directly implement, machine learning is required. Humans are constrained by our inability to manually access vast amounts of data; as a result, we require computer systems, which is where machine learning comes in to simplify our lives.

By providing them with a large amount of data and allowing them to automatically explore the data, build models, and predict the required output, we can train machine learning algorithms. The cost function can be used to determine the amount of data and the machine learning algorithm's performance. We can save both time and money by using machine learning.

The significance of AI can be handily perceived by its utilization's cases, Presently, AI is utilized in self-driving vehicles, digital misrepresentation identification, face acknowledgment, and companion idea by Facebook, and so on. Different top organizations, for example, Netflix and Amazon have constructed AI models that are utilizing an immense measure of information to examine the client interest and suggest item likewise.

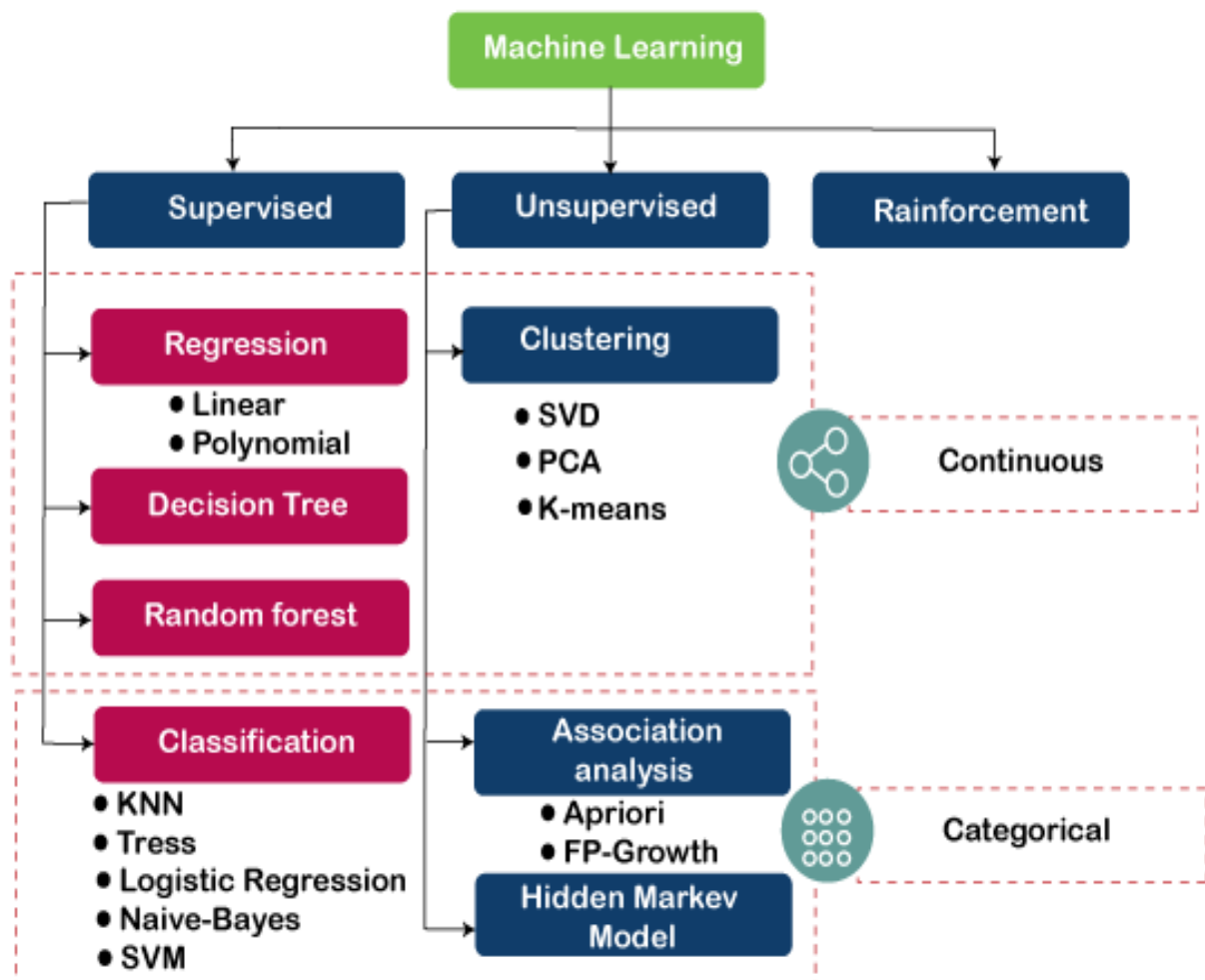
Following are some key points which show the importance of Machine Learning:

- Rapid increment in the production of data
- Solving complex problems, which are difficult for a human
- Decision making in various sector including finance
- Finding hidden patterns and extracting useful information from data.

# Classification of Machine Learning

At a broad level, machine learning can be classified into three types:

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning



## 1) Supervised Learning

In supervised learning, sample labelled data are provided to the machine learning system for training, and the system then predicts the output based on the training data.

The system uses labelled data to build a model that understands the datasets and learns about each one. After the training and processing are done, we test the model with sample data to see if it can accurately predict the output.

The mapping of the input data to the output data is the objective of supervised learning. The managed learning depends on oversight, and it is equivalent to when an

understudy learns things in the management of the educator. Spam filtering is an example of supervised learning.

Supervised learning can be grouped further in two categories of algorithms:

- **Classification**
- **Regression**

## 2) Unsupervised Learning

Unsupervised learning is a learning method in which a machine learns without any supervision.

The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data. It can be further classified into two categories of algorithms:

- **Clustering**
- **Association**

## 3) Reinforcement Learning

Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance. **Q-Learning algorithm** is used in reinforcement learning.

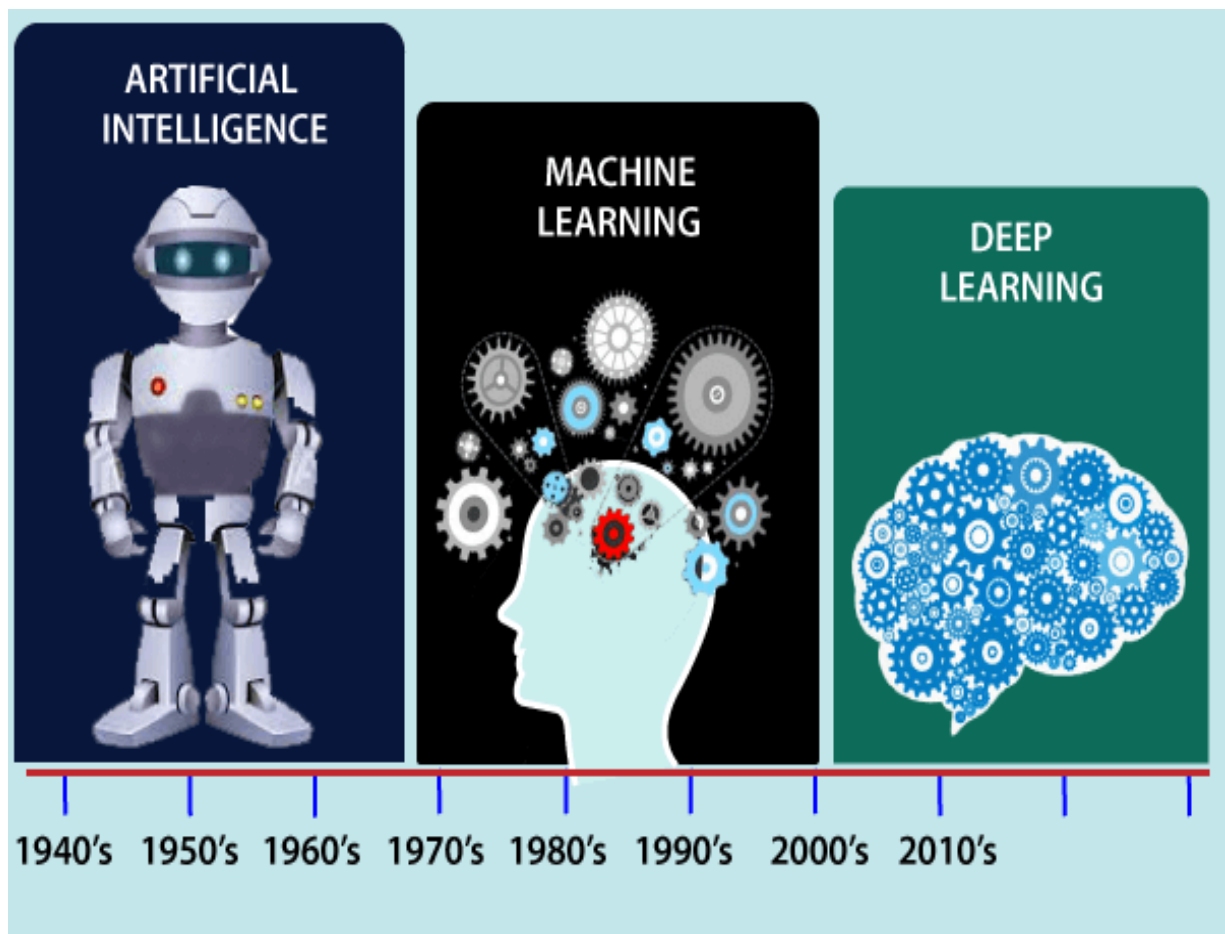
The robotic dog, which automatically learns the movement of his arms, is an example of Reinforcement learning.

**Note:** We will learn about the above types of machine learning in detail in later chapters.

## History of Machine Learning

Before some years (about 40-50 years), machine learning was science fiction, but today it is the part of our daily life. Machine learning is making our day to day life easy from **self-driving cars** to **Amazon virtual assistant "Alexa"**. However, the idea

behind machine learning is so old and has a long history. Below some milestones are given which have occurred in the history of machine learning:



### The early history of Machine Learning (Pre-1940):

- **1834:** In 1834, Charles Babbage, the father of the computer, conceived a device that could be programmed with punch cards. However, the machine was never built, but all modern computers rely on its logical structure.
- **1936:** In 1936, Alan Turing gave a theory that how a machine can determine and execute a set of instructions.

### The era of stored program computers:

- **1940:** In 1940, the first manually operated computer, "ENIAC" was invented, which was the first electronic general-purpose computer. After that stored program computer such as EDSAC in 1949 and EDVAC in 1951 were invented.
- **1943:** In 1943, a human neural network was modeled with an electrical circuit. In 1950, the scientists started applying their idea to work and analyzed how human neurons might work.

## Computer machinery and intelligence:

- **1950:** In 1950, Alan Turing published a seminal paper, "**Computer Machinery and Intelligence**," on the topic of artificial intelligence. **In his paper, he asked, "Can machines think?"**

## Machine intelligence in Games:

- **1952:** Arthur Samuel, who was the pioneer of machine learning, created a program that helped an IBM computer to play a checkers game. It performed better more it played.
- **1959:** In 1959, the term "Machine Learning" was first coined by **Arthur Samuel**.

## The first "AI" winter:

- The duration of 1974 to 1980 was the tough time for AI and ML researchers, and this duration was called as **AI winter**.
- In this duration, failure of machine translation occurred, and people had reduced their interest from AI, which led to reduced funding by the government to the researches.

## Machine Learning from theory to reality

- **1959:** In 1959, the first neural network was applied to a real-world problem to remove echoes over phone lines using an adaptive filter.
- **1985:** In 1985, Terry Sejnowski and Charles Rosenberg invented a neural network **NETtalk**, which was able to teach itself how to correctly pronounce 20,000 words in one week.
- **1997:** The IBM's **Deep blue** intelligent computer won the chess game against the chess expert Garry Kasparov, and it became the first computer which had beaten a human chess expert.

## Machine Learning at 21st century

### 2006:

- Geoffrey Hinton and his group presented the idea of profound getting the hang of utilizing profound conviction organizations.
- The Elastic Compute Cloud (EC2) was launched by Amazon to provide scalable computing resources that made it easier to create and implement machine learning models.

**2007:**

- Participants were tasked with increasing the accuracy of Netflix's recommendation algorithm when the Netflix Prize competition began.
- Support learning made critical progress when a group of specialists utilized it to prepare a PC to play backgammon at a top-notch level.

**2008:**

- Google delivered the Google Forecast Programming interface, a cloud-based help that permitted designers to integrate AI into their applications.
- Confined Boltzmann Machines (RBMs), a kind of generative brain organization, acquired consideration for their capacity to demonstrate complex information conveyances.

**2009:**

- Profound learning gained ground as analysts showed its viability in different errands, including discourse acknowledgment and picture grouping.
- The expression "Large Information" acquired ubiquity, featuring the difficulties and open doors related with taking care of huge datasets.

**2010:**

- The ImageNet Huge Scope Visual Acknowledgment Challenge (ILSVRC) was presented, driving progressions in PC vision, and prompting the advancement of profound convolutional brain organizations (CNNs).

**2011:**

- On Jeopardy! IBM's Watson defeated human champions., demonstrating the potential of question-answering systems and natural language processing.

**2012:**

- AlexNet, a profound CNN created by Alex Krizhevsky, won the ILSVRC, fundamentally further developing picture order precision and laying out profound advancing as a predominant methodology in PC vision.
- Google's Cerebrum project, drove by Andrew Ng and Jeff Dignitary, utilized profound figuring out how to prepare a brain organization to perceive felines from unlabeled YouTube recordings.

**2013:**

- Ian Goodfellow introduced generative adversarial networks (GANs), which made it possible to create realistic synthetic data.
- Google later acquired the startup DeepMind Technologies, which focused on deep learning and artificial intelligence.

**2014:**

- Facebook presented the DeepFace framework, which accomplished close human precision in facial acknowledgment.
- AlphaGo, a program created by DeepMind at Google, defeated a world champion Go player and demonstrated the potential of reinforcement learning in challenging games.

**2015:**

- Microsoft delivered the Mental Toolbox (previously known as CNTK), an open-source profound learning library.
- The performance of sequence-to-sequence models in tasks like machine translation was enhanced by the introduction of the idea of attention mechanisms.

**2016:**

- The goal of explainable AI, which focuses on making machine learning models easier to understand, received some attention.
- Google's DeepMind created AlphaGo Zero, which accomplished godlike Go abilities to play without human information, utilizing just support learning.

**2017:**

- Move learning acquired noticeable quality, permitting pretrained models to be utilized for different errands with restricted information.
- Better synthesis and generation of complex data were made possible by the introduction of generative models like variational autoencoders (VAEs) and Wasserstein GANs.
- These are only a portion of the eminent headways and achievements in AI during the predefined period. The field kept on advancing quickly past 2017, with new leap forwards, strategies, and applications arising.



## **Machine Learning at present:**

The field of machine learning has made significant strides in recent years, and its applications are numerous, including self-driving cars, Amazon Alexa, Catboats, and the recommender system. It incorporates clustering, classification, decision tree, SVM algorithms, and reinforcement learning, as well as unsupervised and supervised learning.

Present day AI models can be utilized for making different expectations, including climate expectation, sickness forecast, financial exchange examination, and so on.

## **Prerequisites**

Before learning machine learning, you must have the basic knowledge of followings so that you can easily understand the concepts of machine learning:

- Fundamental knowledge of probability and linear algebra.
- The ability to code in any computer language, especially in Python language.
- Knowledge of Calculus, especially derivatives of single variable and multivariate functions.

## **Audience**

Our Machine learning tutorial is designed to help beginner and professionals.

## **Problems**

We assure you that you will not find any difficulty while learning our Machine learning tutorial. But if there is any mistake in this tutorial, kindly post the problem or error in the contact form so that we can improve it.

## **Applications of Machine learning**

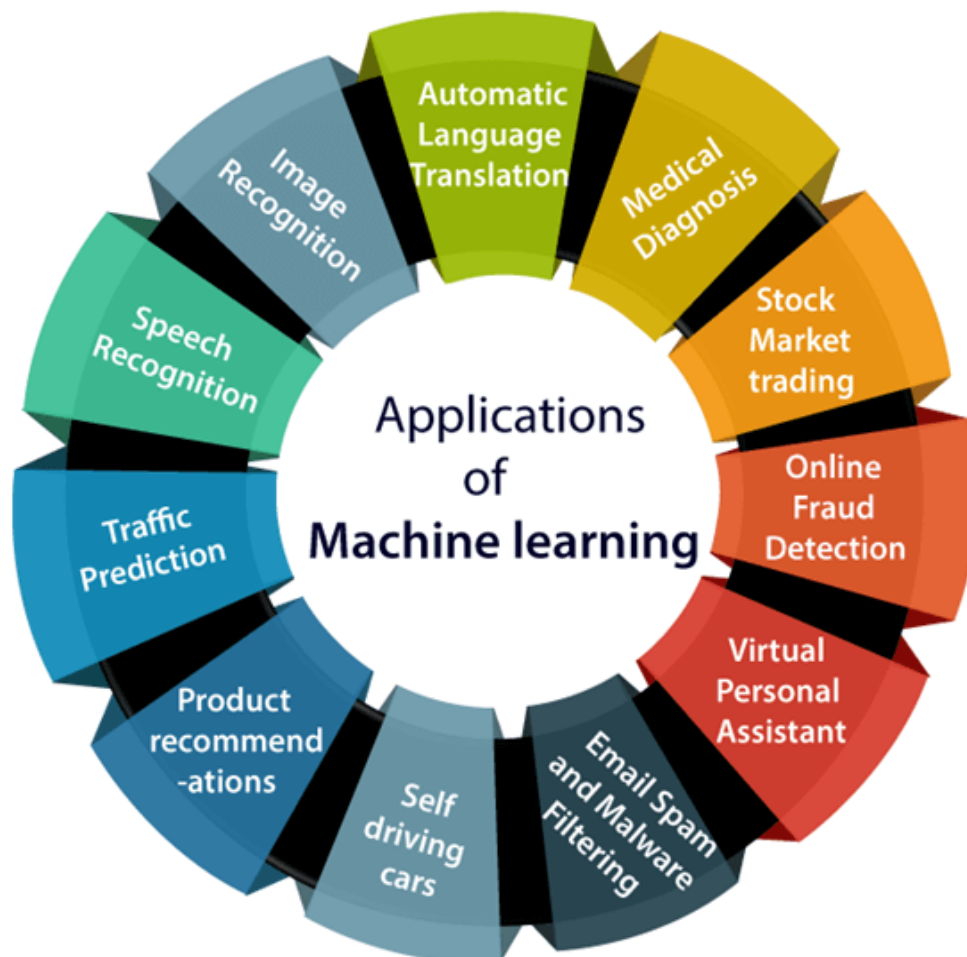
Machine learning is a buzzword for today's technology, and it is growing very rapidly day by day. We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc. Below are some most trending real-world applications of Machine Learning:

### **1. Image Recognition:**

Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, **Automatic friend tagging suggestion**:

Facebook provides us a feature of auto friend tagging suggestion. Whenever we upload a photo with our Facebook friends, then we automatically get a tagging suggestion with name, and the technology behind this is machine learning's **face detection** and **recognition algorithm**.

It is based on the Facebook project named "**Deep Face**," which is responsible for face recognition and person identification in the picture.



## 2. Speech Recognition

While using Google, we get an option of "**Search by voice**," it comes under speech recognition, and it's a popular application of machine learning.

Speech recognition is a process of converting voice instructions into text, and it is also known as "**Speech to text**", or "**Computer speech recognition**." At present, machine learning algorithms are widely used by various applications of speech recognition. **Google assistant, Siri, Cortana, and Alexa** are using speech recognition technology to follow the voice instructions.

### 3. Traffic prediction:

If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions.

It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

- **Real Time location** of the vehicle from Google Map app and sensors
- **Average time has taken** on past days at the same time.

Everyone who is using Google Map is helping this app to make it better. It takes information from the user and sends back to its database to improve the performance.

### 4. Product recommendations:

Machine learning is widely used by various e-commerce and entertainment companies such as **Amazon**, **Netflix**, etc., for product recommendation to the user. Whenever we search for some product on Amazon, then we started getting an advertisement for the same product while internet surfing on the same browser and this is because of machine learning.

Google understands the user interest using various machine learning algorithms and suggests the product as per customer interest.

As similar, when we use Netflix, we find some recommendations for entertainment series, movies, etc., and this is also done with the help of machine learning.

### 5. Self-driving cars:

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

### 6. Email Spam and Malware Filtering:

Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning. Below are some spam filters used by Gmail:

- Content Filter
- Header filter

- General blacklists filter
- Rules-based filters
- Permission filters

Some machine learning algorithms such as **Multi-Layer Perceptron**, **Decision tree**, and **Naïve Bayes classifier** are used for email spam filtering and malware detection.

## 7. Virtual Personal Assistant:

We have various virtual personal assistants such as **Google assistant**, **Alexa**, **Cortana**, **Siri**. As the name suggests, they help us in finding the information using our voice instruction. These assistants can help us in various ways just by our voice instructions such as Play music, call someone, Open an email, Scheduling an appointment, etc.

These virtual assistants use machine learning algorithms as an important part.

These assistant record our voice instructions, send it over the server on a cloud, and decode it using ML algorithms and act accordingly.

## 8. Online Fraud Detection:

Machine learning is making our online transaction safe and secure by detecting fraud transaction. Whenever we perform some online transaction, there may be various ways that a fraudulent transaction can take place such as **fake accounts**, **fake ids**, and **steal money** in the middle of a transaction. So to detect this, **Feed Forward Neural network** helps us by checking whether it is a genuine transaction or a fraud transaction.

For each genuine transaction, the output is converted into some hash values, and these values become the input for the next round. For each genuine transaction, there is a specific pattern which gets change for the fraud transaction hence, it detects it and makes our online transactions more secure.

## 9. Stock Market trading:

Machine learning is widely used in stock market trading. In the stock market, there is always a risk of up and downs in shares, so for this machine learning's **long short term memory neural network** is used for the prediction of stock market trends.

## 10. Medical Diagnosis:

In medical science, machine learning is used for diseases diagnoses. With this, medical technology is growing very fast and able to build 3D models that can predict

the exact position of lesions in the brain. It helps in finding brain tumors and other brain-related diseases easily.

## **11. Automatic Language Translation:**

Nowadays, if we visit a new place and we are not aware of the language then it is not a problem at all, as for this also machine learning helps us by converting the text into our known languages. Google's GNMT (Google Neural Machine Translation) provide this feature, which is a Neural Machine Learning that translates the text into our familiar language, and it called as automatic translation.

The technology behind the automatic translation is a sequence to sequence learning algorithm, which is used with image recognition and translates the text from one language to another language.

## **Machine learning Life cycle**

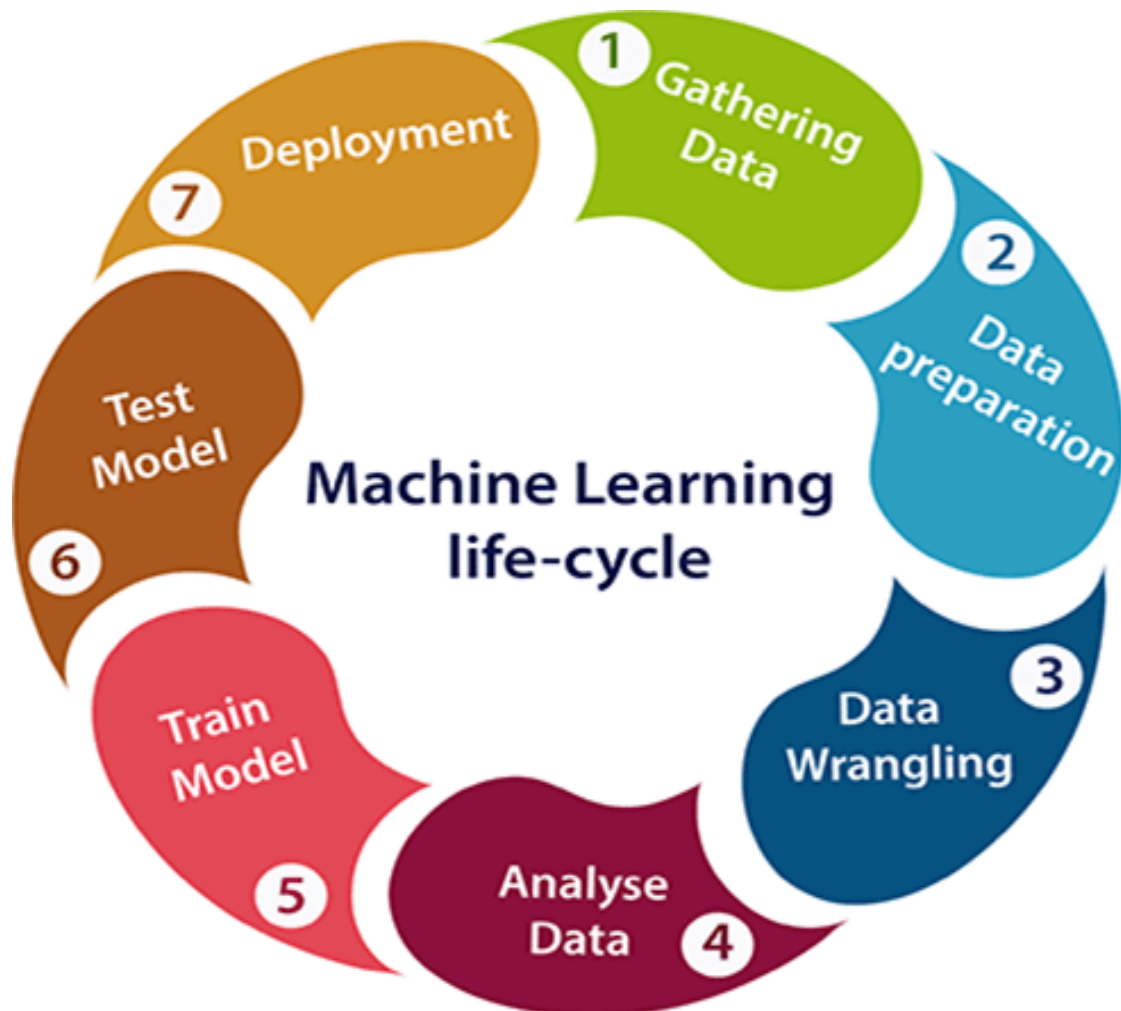
Machine learning has given the computer systems the abilities to automatically learn without being explicitly programmed. But how does a machine learning system work? So, it can be described using the life cycle of machine learning. Machine learning life cycle is a cyclic process to build an efficient machine learning project. The main purpose of the life cycle is to find a solution to the problem or project.

Machine learning life cycle involves seven major steps, which are given below:

- 1. Gathering Data**
- 2. Data preparation**
- 3. Data Wrangling**
- 4. Analyse Data**
- 5. Train the model**
- 6. Test the model**
- 7. Deployment**

The most important thing in the complete process is to understand the problem and to know the purpose of the problem. Therefore, before starting the life cycle, we need to understand the problem because the good result depends on the better understanding of the problem.

In the complete life cycle process, to solve a problem, we create a machine learning system called "model", and this model is created by providing "training". But to train a model, we need data, hence, life cycle starts by collecting data.



## 1. Gathering Data:

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.

In this step, we need to identify the different data sources, as data can be collected from various sources such as **files**, **database**, **internet**, or **mobile devices**. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

This step includes the below tasks:

- **Identify various data sources**

- **Collect data**
- **Integrate the data obtained from different sources**

By performing the above task, we get a coherent set of data, also called as a **dataset**. It will be used in further steps.

## **2. Data preparation**

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

- **Data exploration:**  
It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data.  
A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.
- **Data pre-processing:**  
Now the next step is preprocessing of data for its analysis.

## **3. Data Wrangling**

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful. In real-world applications, collected data may have various issues, including:

- **Missing Values**
- **Duplicate data**
- **Invalid data**
- **Noise**

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

## 4. Data Analysis

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

- **Selection of analytical techniques**
- **Building models**
- **Review the result**

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as **Classification, Regression, Cluster analysis, Association**, etc. then build the model using prepared data, and evaluate the model.

Hence, in this step, we take the data and use machine learning algorithms to build the model.

## 5. Train Model

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem.

We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features.

## 6. Test Model

Once our machine learning model has been trained on a given dataset, then we test the model. In this step, we check for the accuracy of our model by providing a test dataset to it.

Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

## 7. Deployment

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.



If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project.

## **Difference between Artificial intelligence and Machine learning**

Artificial intelligence and machine learning are the part of computer science that are correlated with each other. These two technologies are the most trending technologies which are used for creating intelligent systems.

Although these are two related technologies and sometimes people use them as a synonym for each other, but still, both are the two different terms in various cases.

On a broad level, we can differentiate both AI and ML as:

***AI is a bigger concept to create intelligent machines that can simulate human thinking capability and behaviour, whereas, machine learning is an application or subset of AI that allows machines to learn from data without being programmed explicitly.***

Below are some main differences between AI and machine learning along with the overview of Artificial intelligence and machine learning.

## **Artificial Intelligence**

Artificial intelligence is a field of computer science which makes a computer system that can mimic human intelligence. It is comprised of two words "**Artificial**" and "**intelligence**", which means "a human-made thinking power." Hence we can define it as,

***Artificial intelligence is a technology using which we can create intelligent systems that can simulate human intelligence.***

The Artificial intelligence system does not require to be pre-programmed, instead of that, they use such algorithms which can work with their own intelligence. It involves machine learning algorithms such as Reinforcement learning algorithm and deep learning neural networks. AI is being used in multiple places such as Siri, Google's AlphaGo, AI in Chess playing, etc.

Based on capabilities, AI can be classified into three types:

- **Weak AI**
- **General AI**

- **Strong AI**

Currently, we are working with weak AI and general AI. The future of AI is Strong AI for which it is said that it will be intelligent than humans.

## **Machine learning**

Machine learning is about extracting knowledge from the data. It can be defined as,

***Machine learning is a subfield of artificial intelligence, which enables machines to learn from past data or experiences without being explicitly programmed.***

Machine learning enables a computer system to make predictions or take some decisions using historical data without being explicitly programmed. Machine learning uses a massive amount of structured and semi-structured data so that a machine learning model can generate accurate result or give predictions based on that data.

Machine learning works on algorithm which learn by it's own using historical data. It works only for specific domains such as if we are creating a machine learning model to detect pictures of dogs, it will only give result for dog images, but if we provide a new data like cat image then it will become unresponsive. Machine learning is being used in various places such as for online recommender system, for Google search algorithms, Email spam filter, Facebook Auto friend tagging suggestion, etc.

It can be divided into three types:

- **Supervised learning**
- **Reinforcement learning**
- **Unsupervised learning**

## **Key differences between Artificial Intelligence (AI) and Machine learning (ML):**

<b>Artificial Intelligence</b>	<b>Machine learning</b>
<b>Artificial intelligence is a technology which enables a machine to simulate human behaviour.</b>	Machine learning is a subset of AI which allows a machine to automatically learn from past data without programming explicitly.
The goal of AI is to make a smart computer system like humans to solve complex problems.	The goal of ML is to allow machines to learn from data so that they can give accurate output.

In AI, we make intelligent systems to perform any task like a human.	In ML, we teach machines with data to perform a particular task and give an accurate result.
Machine learning and deep learning are the two main subsets of AI.	Deep learning is a main subset of machine learning.
AI has a very wide range of scope.	Machine learning has a limited scope.
AI is working to create an intelligent system which can perform various complex tasks.	Machine learning is working to create machines that can perform only those specific tasks for which they are trained.
AI system is concerned about maximizing the chances of success.	Machine learning is mainly concerned about accuracy and patterns.
The main applications of AI are Siri, customer support using chatbots, Expert System, Online game playing, intelligent humanoid robot, etc.	The main applications of machine learning are Online recommender system, Google search algorithms, Facebook auto friend tagging suggestions, etc.
On the basis of capabilities, AI can be divided into three types, which are, Weak AI, General AI, and Strong AI.	Machine learning can also be divided into mainly three types that are Supervised learning, Unsupervised learning, and Reinforcement learning.
It includes learning, reasoning, and self-correction.	It includes learning and self-correction when introduced with new data.
AI completely deals with Structured, semi-structured, and unstructured data.	Machine learning deals with Structured and semi-structured data.

## How to get datasets for Machine Learning

The field of ML depends vigorously on datasets for preparing models and making precise predictions. Datasets assume a vital part in the progress of AIML projects and are fundamental for turning into a gifted information researcher. In this article, we will investigate the various sorts of datasets utilized in AI and give a definite aid on where to track down them.

### What is a dataset?

**A dataset** is a collection of data in which data is arranged in some order. A dataset can contain any data from a series of an array to a database table. Below table shows an example of the dataset:

A tabular dataset can be understood as a database table or matrix, where each column corresponds to a **particular variable**, and each row corresponds to the **fields of the dataset**. The most supported file type for a tabular dataset is "**Comma Separated File**," or **CSV**. But to store a "tree-like data," we can use the JSON file more efficiently.

## Types of data in datasets

- **Numerical data:**Such as house price, temperature, etc.
- **Categorical data:**Such as Yes/No, True/False, Blue/green, etc.
- **Ordinal data:**These data are similar to categorical data but can be measured on the basis of comparison.

Note: A real-world dataset is of huge size, which is difficult to manage and process at the initial level. Therefore, to practice machine learning algorithms, we can use any dummy dataset.

## Types of datasets

Machine learning incorporates different domains, each requiring explicit sorts of datasets. A few normal sorts of datasets utilized in machine learning include:

### Image Datasets:

Image datasets contain an assortment of images and are normally utilized in computer vision tasks such as image classification, object detection, and image segmentation.

#### Examples :

- ImageNet
- CIFAR-10
- MNIST

### Text Datasets:

Text datasets comprise textual information, like articles, books, or virtual entertainment posts. These datasets are utilized in NLP techniques like sentiment analysis, text classification, and machine translation.

#### Examples :

- Gutenberg Task dataset
- IMDb film reviews dataset

## Time Series Datasets:

Time series datasets include information focuses gathered after some time. They are generally utilized in determining, abnormality location, and pattern examination. **Examples :**

- Securities exchange information
- Climate information
- Sensor readings.

## Tabular Datasets:

Tabular datasets are organized information coordinated in tables or calculation sheets. They contain lines addressing examples or tests and segments addressing highlights or qualities. Tabular datasets are utilized for undertakings like relapse and arrangement. The dataset given before in the article is an illustration of a tabular dataset.

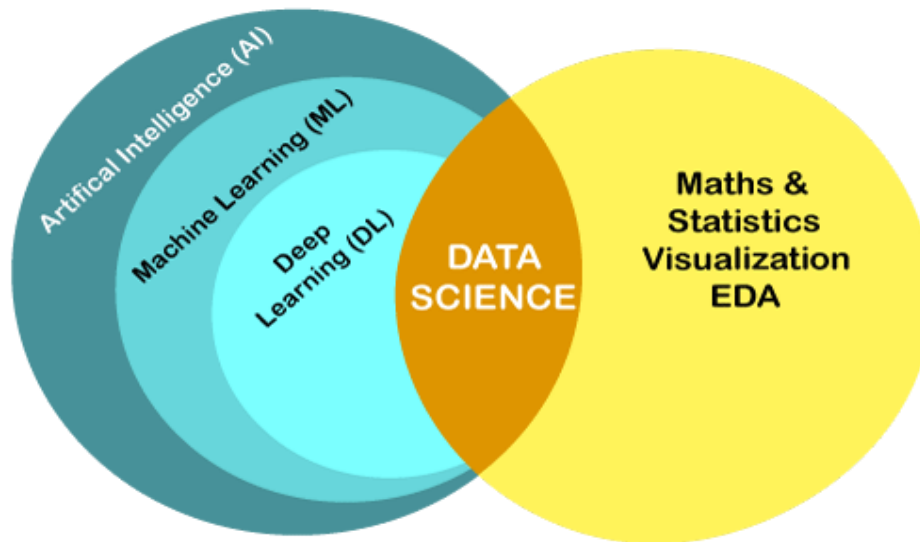
## Need of Dataset

- Completely ready and pre-handled datasets are significant for machine learning projects.
- They give the establishment to prepare exact and solid models. Notwithstanding, working with enormous datasets can introduce difficulties regarding the board and handling.

## Difference Between Data Science and Machine Learning

Data Science is the study of *data cleansing, preparation, and analysis*, while machine learning is a branch of AI and subfield of data science. Data Science and Machine Learning are the two popular modern technologies, and they are growing with an immoderate rate. But these two buzzwords, along with artificial intelligence and deep learning are very confusing term, so it is important to understand how they are different from each other. In this topic, we will understand the difference between Data Science and Machine Learning only, and how they relate to each other.

Data Science and Machine Learning are closely related to each other but have different functionalities and different goals. At a glance, *Data Science is a field to study the approaches to find insights from the raw data. Whereas, Machine Learning is a technique used by the group of data scientists to enable the machines to learn automatically from the past data.* To understand the difference in-depth, let's first have a brief introduction to these two technologies.



Note: Data Science and Machine Learning are closely related to each other but cannot be treated as synonyms.

## What is Data Science?

Data science, as its name suggests, is all about the data. Hence, we can define it as, ***"A field of deep study of data that includes extracting useful insights from the data, and processing that information using different tools, statistical models, and Machine learning algorithms."*** It is a concept that is used to handle big data that includes data cleaning, data preparation, data analysis, and data visualization.

A data scientist collects the raw data from various sources, prepares and pre-processes the data, and applies machine learning algorithms, predictive analysis to extract useful insights from the collected data.

For example, Netflix uses data science techniques to understand user interest by mining the data and viewing patterns of its users.

## Skills Required to become Data Scientist

- An excellent programming knowledge of **Python, R, SAS, or Scala**.
- Experience in SQL database Coding.
- Knowledge of Machine Learning Algorithms.
- Deep Knowledge of Statistics concepts.
- Data Mining, cleaning, and Visualizing skills.
- Skills to use Big data tools such as Hadoop.

## What is Machine Learning?

Machine learning is a part of artificial intelligence and the subfield of Data Science. It is a growing technology that enables machines to learn from past data and perform a given task automatically. It can be defined as:

Machine Learning allows the computers to learn from the past experiences by its own, it uses statistical methods to improve the performance and predict the output without being explicitly programmed.

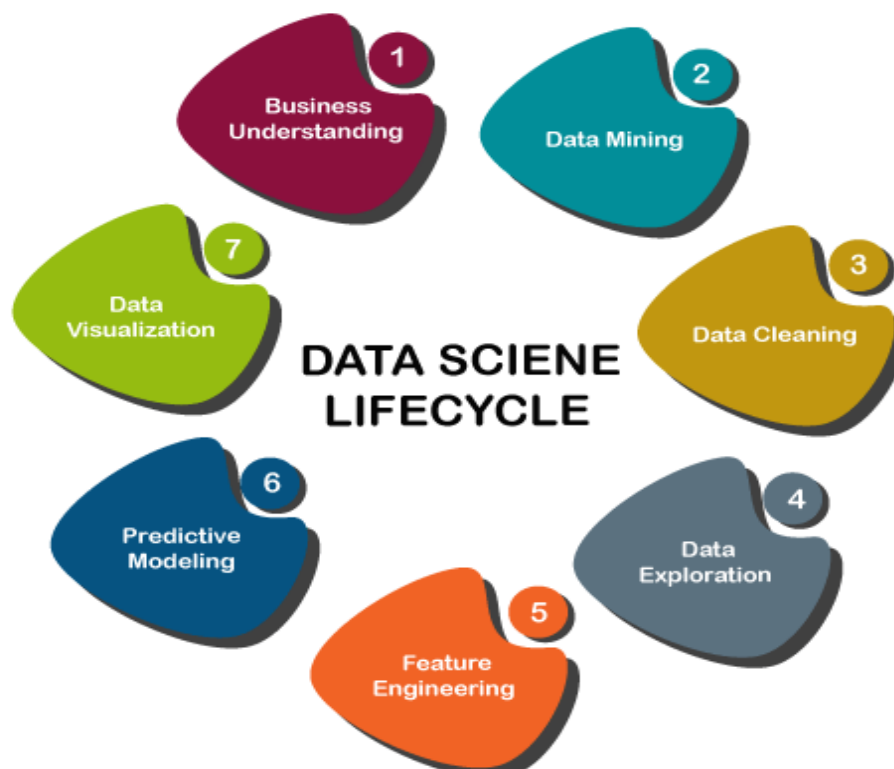
The popular applications of ML are **Email spam filtering, product recommendations, online fraud detection, etc.**

### Skills Needed for the Machine Learning Engineer:

- Understanding and implementation of Machine Learning Algorithms.
- Natural Language Processing.
- Good Programming knowledge of Python or R.
- Knowledge of Statistics and probability concepts.
- Knowledge of data modeling and data evaluation.

## Where is Machine Learning used in Data Science?

The use of machine learning in data science can be understood by the development process or life cycle of Data Science. The different steps that occur in Data science lifecycle are as follows:



1. **Business Requirements:** In this step, we try to understand the requirement for the business problem for which we want to use it. Suppose we want to create a recommendation system, and the business requirement is to increase sales.
2. **Data Acquisition:** In this step, the data is acquired to solve the given problem. For the recommendation system, we can get the ratings provided by the user for different products, comments, purchase history, etc.
3. **Data Processing:** In this step, the raw data acquired from the previous step is transformed into a suitable format, so that it can be easily used by the further steps.
4. **Data Exploration:** It is a step where we understand the patterns of the data, and try to find out the useful insights from the data.
5. **Modeling:** The data modeling is a step where machine learning algorithms are used. So, this step includes the whole machine learning process. The machine learning process involves importing the data, data cleaning, building a model, training the model, testing the model, and improving the model's efficiency.
6. **Deployment & Optimization:** This is the last step where the model is deployed on an actual project, and the performance of the model is checked.

## Comparison Between Data Science and Machine Learning

The below table describes the basic differences between Data Science and ML:

Data Science	Machine Learning
It deals with understanding and finding hidden patterns or useful insights from the data, which helps to take smarter business decisions.	It is a subfield of data science that enables the machine to learn from the past data and experiences automatically.
It is used for discovering insights from the data.	It is used for making predictions and classifying the result for new data points.
It is a broad term that includes various steps to create a model for a given problem and deploy the model.	It is used in the data modeling step of the data science as a complete process.
A data scientist needs to have skills to use big data tools like Hadoop, Hive and Pig, statistics, programming in Python, R, or Scala.	Machine Learning Engineer needs to have skills such as computer science fundamentals, programming skills in Python or R, statistics and probability concepts, etc.

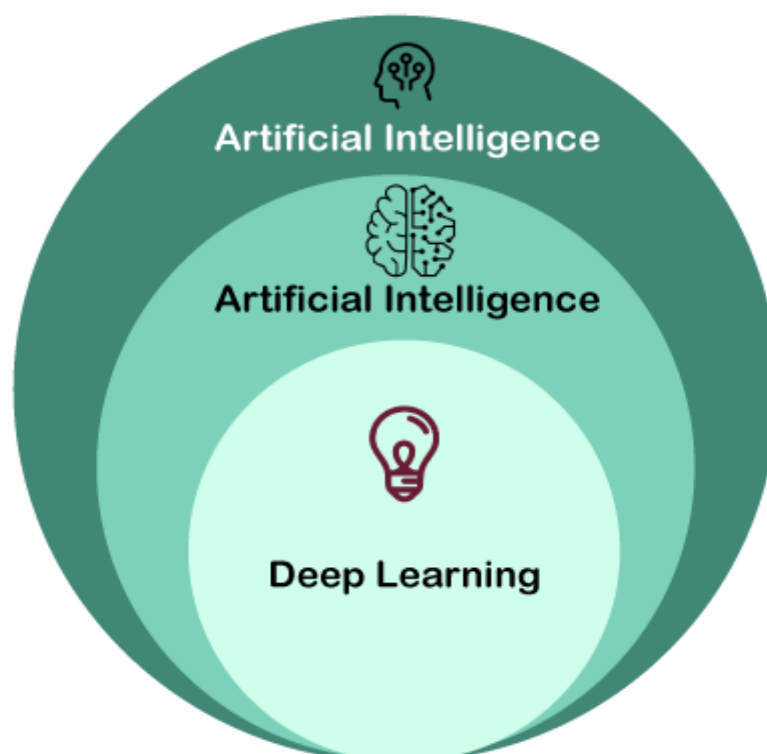


It can work with raw, structured, and unstructured data.	It mostly requires structured data to work on.
Data scientists spent lots of time in handling the data, cleansing the data, and understanding its patterns.	ML engineers spend a lot of time for managing the complexities that occur during the implementation of algorithms and mathematical concepts behind that.

## Difference between Machine Learning and Deep Learning

Machine Learning and Deep Learning are the two main concepts of Data Science and the subsets of Artificial Intelligence. Most of the people think the machine learning, deep learning, and as well as artificial intelligence as the same buzzwords. But in actuality, all these terms are different but related to each other.

In this topic, we will learn how machine learning is different from deep learning. But before learning the differences, let's first have a brief introduction of machine learning and deep learning.



## What is Machine Learning?

Machine learning is a part of artificial intelligence and growing technology that enables machines to learn from past data and perform a given task automatically.

Machine Learning allows the computers to learn from the experiences by its own, use statistical methods to improve the performance and predict the output without being explicitly programmed.

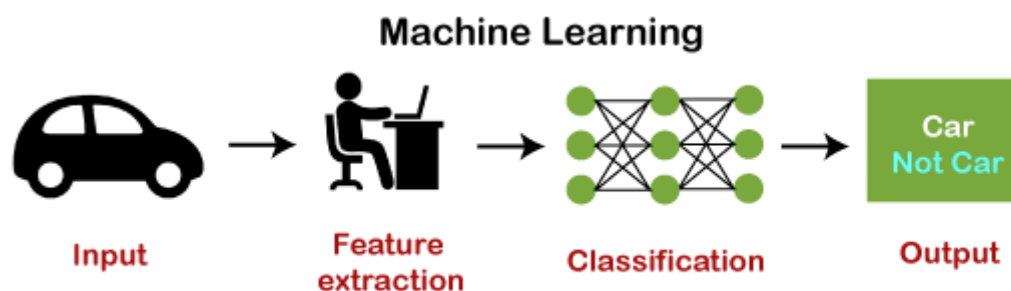
The popular applications of ML are Email spam filtering, product recommendations, online fraud detection, etc.

Some useful ML algorithms are:

- **Decision Tree algorithm**
- **Naïve Bayes**
- **Random Forest**
- **K-means clustering**
- **KNN algorithm**
- **Apriori Algorithm, etc.**

## How does Machine Learning work?

The working of machine learning models can be understood by the example of identifying the image of a cat or dog. To identify this, the ML model takes images of both cat and dog as input, extracts the different features of images such as shape, height, nose, eyes, etc., applies the classification algorithm, and predict the output. Consider the below image:



## What is Deep Learning?

*Deep Learning is the subset of machine learning or can be said as a special kind of machine learning.* It works technically in the same way as machine learning does, but with different capabilities and approaches. It is inspired by the functionality of human brain cells, which are called neurons, and leads to the concept of artificial neural networks. It is also called a deep neural network or deep neural learning.

In deep learning, models use different layers to learn and discover insights from the data.

Some popular applications of deep learning are self-driving cars, language translation, natural language processing, etc.

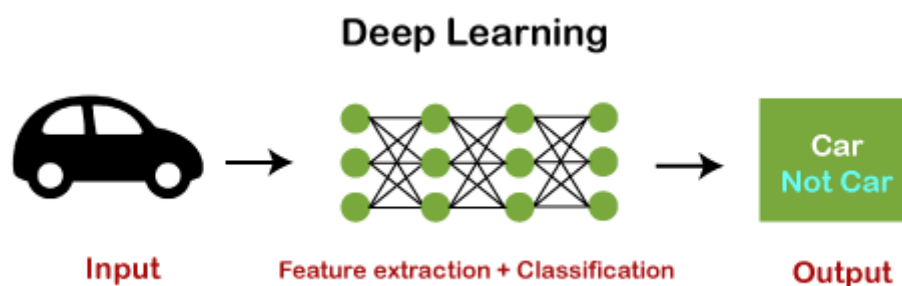
Some popular deep learning models are:

- **Convolutional Neural Network**
- **Recurrent Neural Network**
- **Autoencoders**
- **Classic Neural Networks, etc.**

## How Deep Learning Works?

We can understand the working of deep learning with the same example of identifying cat vs. dog. The deep learning model takes the images as the input and feed it directly to the algorithms without requiring any manual feature extraction step. The images pass to the different layers of the artificial neural network and predict the final output.

Consider the below image:



## Key comparisons between Machine Learning and Deep Learning

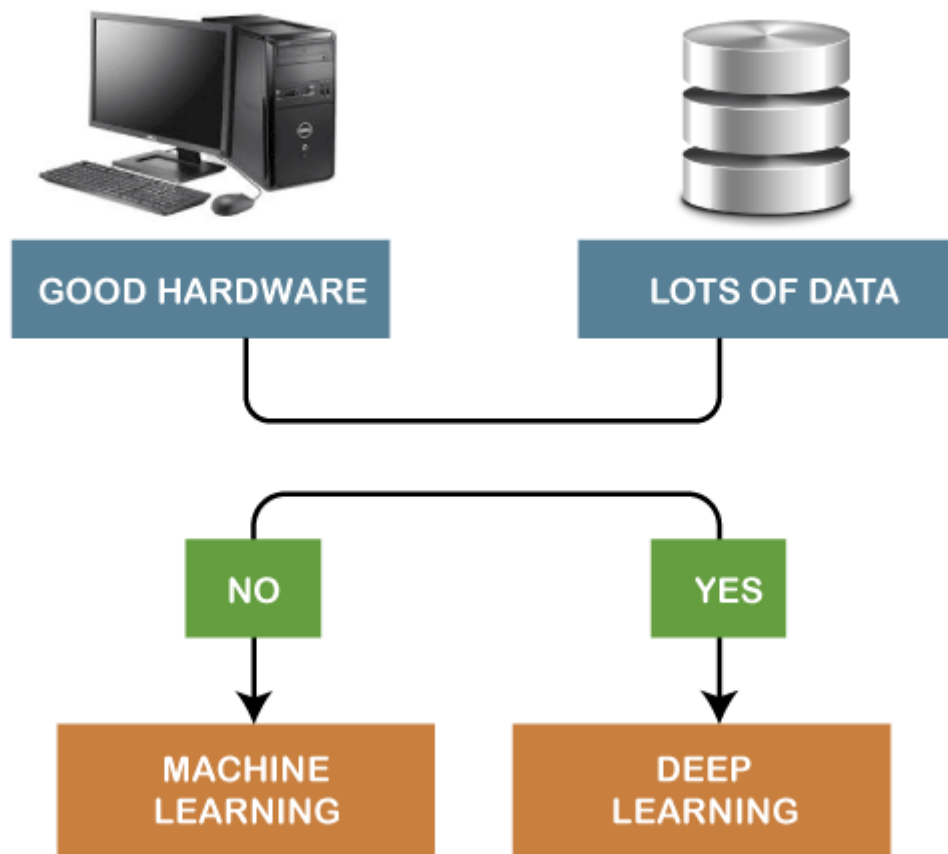
Let's understand the key differences between these two terms based on different parameters:

Parameter	Machine Learning	Deep Learning
<b>Data Dependency</b>	Although machine learning depends on the huge amount of data, it can work with a smaller amount of data.	Deep Learning algorithms highly depend on a large amount of data, so we need to feed a large amount of data for good performance.
<b>Execution time</b>	Machine learning algorithm takes less time to train the model than deep learning, but	Deep Learning takes a long execution time to train the

	it takes a long-time duration to test the model.	model, but less time to test the model.
<b>Hardware Dependencies</b>	Since machine learning models do not need much amount of data, so they can work on low-end machines.	The deep learning model needs a huge amount of data to work efficiently, so they need GPU's and hence the high-end machine.
<b>Feature Engineering</b>	Machine learning models need a step of feature extraction by the expert, and then it proceeds further.	Deep learning is the enhanced version of machine learning, so it does not need to develop the feature extractor for each problem; instead, it tries to learn high-level features from the data on its own.
<b>Problem-solving approach</b>	To solve a given problem, the traditional ML model breaks the problem in sub-parts, and after solving each part, produces the final result.	The problem-solving approach of a deep learning model is different from the traditional ML model, as it takes input for a given problem, and produce the end result. Hence it follows the end-to-end approach.
<b>Interpretation of result</b>	The interpretation of the result for a given problem is easy. As when we work with machine learning, we can interpret the result easily, it means why this result occur, what was the process.	The interpretation of the result for a given problem is very difficult. As when we work with the deep learning model, we may get a better result for a given problem than the machine learning model, but we cannot find why this particular outcome occurred, and the reasoning.
<b>Type of data</b>	Machine learning models mostly require data in a structured form.	Deep Learning models can work with structured and unstructured data both as they rely on the layers of the Artificial neural network.
<b>Suitable for</b>	Machine learning models are suitable for solving simple or bit-complex problems.	Deep learning models are suitable for solving complex problems.

## Which one to select among ML and Deep Learning?

As we have seen the brief introduction of ML and DL with some comparisons, now why and which one needs to be chosen to solve a particular problem. So, it can be understood by the given flowchart:



Hence, if you have lots of data and high hardware capabilities, go with deep learning. But if you don't have any of them, choose the ML model to solve your problem.

**Conclusion:** In conclusion, we can say that deep learning is machine learning with more capabilities and a different working approach. And selecting any of them to solve a particular problem is depend on the amount of data and complexity of the problem.

# TYPES OF MACHINE LEARNING

## 1. Supervised Machine Learning

Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output. The labelled data means some input data is already tagged with the correct output.

In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the supervision of the teacher.

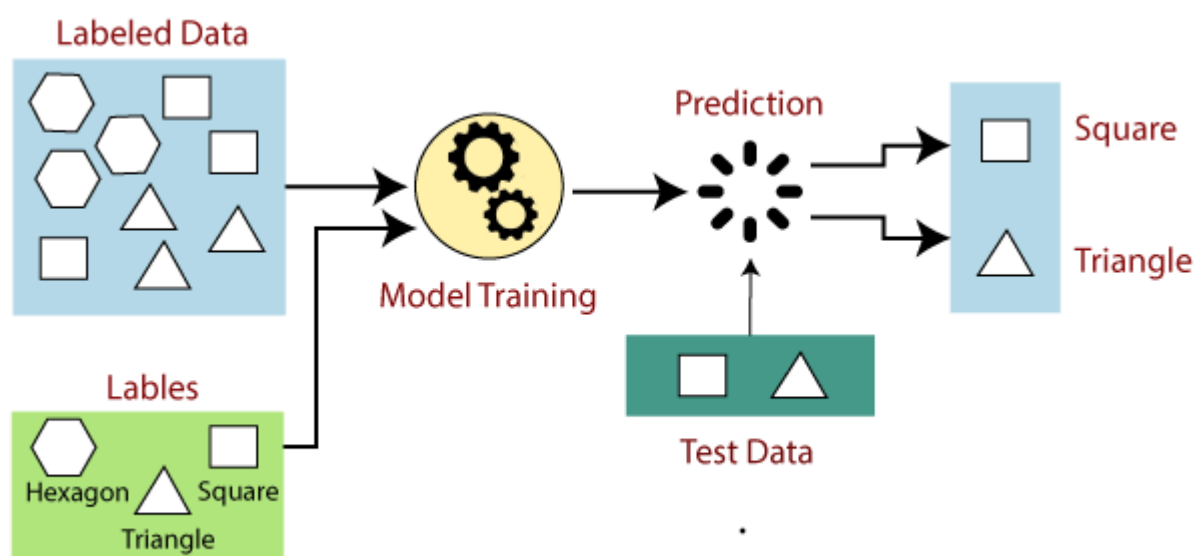
Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to **find a mapping function to map the input variable(x) with the output variable(y)**.

In the real-world, supervised learning can be used for **Risk Assessment, Image classification, Fraud Detection, spam filtering**, etc.

### How Supervised Learning Works?

In supervised learning, models are trained using labelled dataset, where the model learns about each type of data. Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.

The working of Supervised learning can be easily understood by the below example and diagram:



Suppose we have a dataset of different types of shapes which includes square, rectangle, triangle, and Polygon. Now the first step is that we need to train the model for each shape.

- If the given shape has four sides, and all the sides are equal, then it will be labelled as a **Square**.
- If the given shape has three sides, then it will be labelled as a **triangle**.
- If the given shape has six equal sides then it will be labelled as **hexagon**.

Now, after training, we test our model using the test set, and the task of the model is to identify the shape.

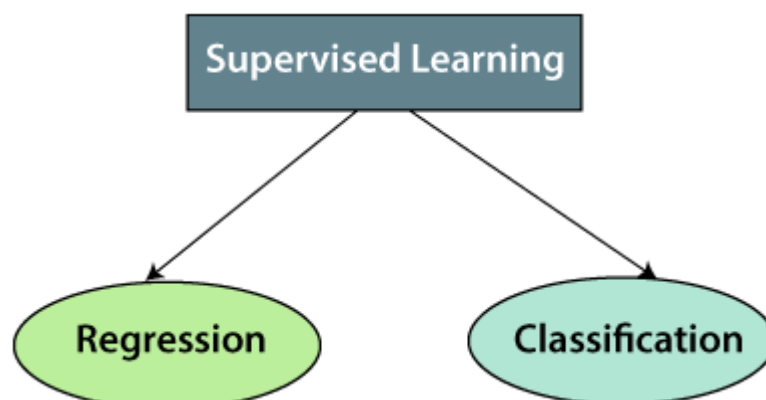
The machine is already trained on all types of shapes, and when it finds a new shape, it classifies the shape on the bases of a number of sides, and predicts the output.

### Steps Involved in Supervised Learning:

- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training **dataset**, **test dataset**, and **validation dataset**.
- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set. If the model predicts the correct output, which means our model is accurate.

### Types of supervised Machine learning Algorithms:

Supervised learning can be further divided into two types of problems:



## 1. Regression

Regression algorithms are used if there is a relationship between the input variable and the output variable. It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc. Below are some popular Regression algorithms which come under supervised learning:

- Linear Regression
- Regression Trees
- Non-Linear Regression
- Bayesian Linear Regression
- Polynomial Regression

## 2. Classification

Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-false, etc.

Spam Filtering,

- Random Forest
- Decision Trees
- Logistic Regression
- Support vector Machines

Note: We will discuss these algorithms in detail in later chapters.

### Advantages of Supervised learning:

- With the help of supervised learning, the model can predict the output on the basis of prior experiences.
- In supervised learning, we can have an exact idea about the classes of objects.
- Supervised learning model helps us to solve various real-world problems such as **fraud detection, spam filtering**, etc.

### Disadvantages of supervised learning:

- Supervised learning models are not suitable for handling the complex tasks.
- Supervised learning cannot predict the correct output if the test data is different from the training dataset.
- Training required lots of computation times.
- In supervised learning, we need enough knowledge about the classes of object.



## Unsupervised Machine Learning

In the previous topic, we learned supervised machine learning in which models are trained using labeled data under the supervision of training data. But there may be many cases in which we do not have labeled data and need to find the hidden patterns from the given dataset. So, to solve such types of cases in machine learning, we need unsupervised learning techniques.

### What is Unsupervised Learning?

As the name suggests, unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things. It can be defined as:

*Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision.*

Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to **find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.**

**Example:** Suppose the unsupervised learning algorithm is given an input dataset containing images of different types of cats and dogs. The algorithm is never trained upon the given dataset, which means it does not have any idea about the features of the dataset. The task of the unsupervised learning algorithm is to identify the image features on their own. Unsupervised learning algorithm will perform this task by clustering the image dataset into the groups according to similarities between images.



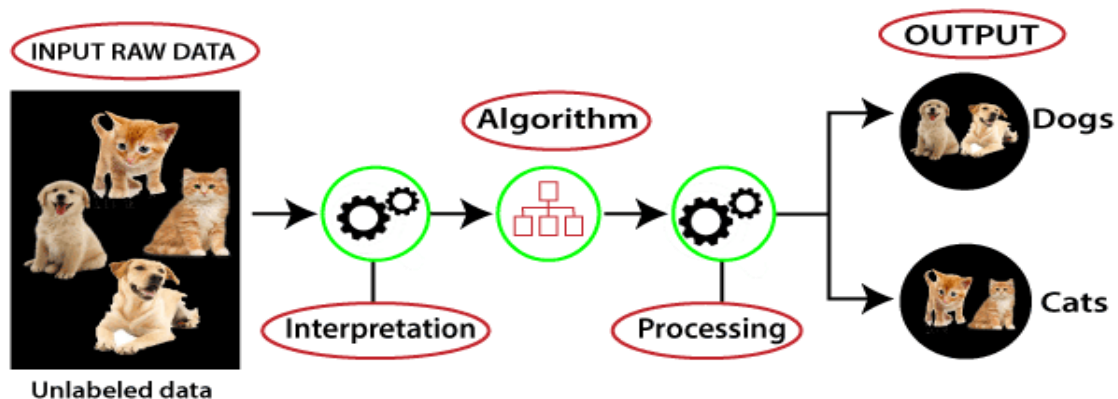
## Why use Unsupervised Learning?

Below are some main reasons which describe the importance of Unsupervised Learning:

- Unsupervised learning is helpful for finding useful insights from the data.
- Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.
- Unsupervised learning works on unlabeled and uncategorized data which make unsupervised learning more important.
- In real-world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.

## Working of Unsupervised Learning

Working of unsupervised learning can be understood by the below diagram:

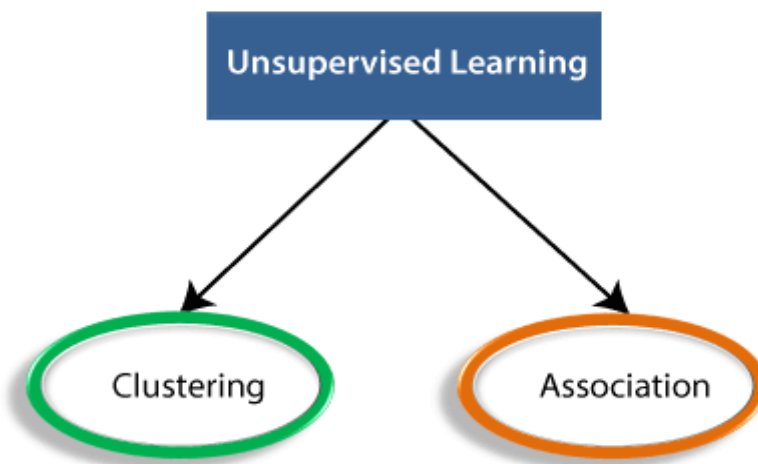


Here, we have taken an unlabeled input data, which means it is not categorized and corresponding outputs are also not given. Now, this unlabeled input data is fed to the machine learning model in order to train it. Firstly, it will interpret the raw data to find the hidden patterns from the data and then will apply suitable algorithms such as k-means clustering, Decision tree, etc.

Once it applies the suitable algorithm, the algorithm divides the data objects into groups according to the similarities and difference between the objects.

## Types of Unsupervised Learning Algorithm:

The unsupervised learning algorithm can be further categorized into two types of problems:



- **Clustering:** Clustering is a method of grouping the objects into clusters such that objects with most similarities remain in a group and have less or no similarities with the objects of another group. Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities.
- **Association:** An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is Market Basket Analysis.

Note: We will learn these algorithms in later chapters.

## Unsupervised Learning algorithms:

Below is the list of some popular unsupervised learning algorithms:

- **K-means clustering**
- **KNN (k-nearest neighbors)**
- **Hierarchical clustering**
- **Anomaly detection**
- **Neural Networks**
- **Principal Component Analysis**
- **Independent Component Analysis**
- **Apriori algorithm**
- **Singular value decomposition**

## Advantages of Unsupervised Learning

- Unsupervised learning is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labeled input data.
- Unsupervised learning is preferable as it is easy to get unlabeled data in comparison to labeled data.

## Disadvantages of Unsupervised Learning

- Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.
- The result of the unsupervised learning algorithm might be less accurate as input data is not labeled, and algorithms do not know the exact output in advance.

## Difference between Supervised and Unsupervised Learning

Supervised and Unsupervised learning are the two techniques of machine learning. But both the techniques are used in different scenarios and with different datasets. Below the explanation of both learning methods along with their difference table is given.

### Supervised Machine Learning:

Supervised learning is a machine learning method in which models are trained using labeled data. In supervised learning, models need to find the mapping function to map the input variable (X) with the output variable (Y).

$$Y = f(X)$$

Supervised learning needs supervision to train the model, which is similar to as a student learns things in the presence of a teacher. Supervised learning can be used for two types of problems: **Classification** and **Regression**.

**Example:** Suppose we have an image of different types of fruits. The task of our supervised learning model is to identify the fruits and classify them accordingly. So to identify the image in supervised learning, we will give the input data as well as output for that, which means we will train the model by the shape, size, color, and taste of each fruit. Once the training is completed, we will test the model by giving the new set of fruit. The model will identify the fruit and predict the output using a suitable algorithm.

## Unsupervised Machine Learning:

Unsupervised learning is another machine learning method in which patterns inferred from the unlabeled input data. The goal of unsupervised learning is to find the structure and patterns from the input data. Unsupervised learning does not need any supervision. Instead, it finds patterns from the data by its own.

Unsupervised learning can be used for two types of problems:

### Clustering and Association.

**Example:** To understand the unsupervised learning, we will use the example given above. So unlike supervised learning, here we will not provide any supervision to the model. We will just provide the input dataset to the model and allow the model to find the patterns from the data. With the help of a suitable algorithm, the model will train itself and divide the fruits into different groups according to the most similar features between them.

The main differences between Supervised and Unsupervised learning are given below:

Supervised Learning	Unsupervised Learning
Supervised learning algorithms are trained using labelled data.	Unsupervised learning algorithms are trained using unlabelled data.
Supervised learning model takes direct feedback to check if it is predicting correct output or not.	Unsupervised learning model does not take any feedback.
Supervised learning model predicts the output.	Unsupervised learning model finds the hidden patterns in data.
In supervised learning, input data is provided to the model along with the output.	In unsupervised learning, only input data is provided to the model.
The goal of supervised learning is to train the model so that it can predict the output when it is given new data.	The goal of unsupervised learning is to find the hidden patterns and useful insights from the unknown dataset.
Supervised learning needs supervision to train the model.	Unsupervised learning does not need any supervision to train the model.
Supervised learning can be categorized in Classification and Regression problems .	Unsupervised Learning can be classified in <b>Clustering</b> and <b>Associations</b> problems.

<b>Supervised learning can be used for those cases where we know the input as well as corresponding outputs.</b>	Unsupervised learning can be used for those cases where we have only input data and no corresponding output data.
<b>Supervised learning model produces an accurate result.</b>	Unsupervised learning model may give less accurate result as compared to supervised learning.
<b>Supervised learning is not close to true Artificial intelligence as in this, we first train the model for each data, and then only it can predict the correct output.</b>	Unsupervised learning is more close to the true Artificial Intelligence as it learns similarly as a child learns daily routine things by his experiences.
<b>It includes various algorithms such as Linear Regression, Logistic Regression, Support Vector Machine, Multi-class Classification, Decision tree, Bayesian Logic, etc.</b>	It includes various algorithms such as Clustering, KNN, and Apriori algorithm.

Note: The supervised and unsupervised learning both are the machine learning methods, and selection of any of these learning depends on the factors related to the structure and volume of your dataset and the use cases of the problem.

## Regression Analysis:

Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values such as **temperature, age, salary, price**, etc.

We can understand the concept of regression analysis using the below example:

**Example:** Suppose there is a marketing company A, who does various advertisement every year and get sales on that. The below list shows the advertisement made by the company in the last 5 years and the corresponding sales:

Advertisement	Sales
\$90	\$1000
\$120	\$1300
\$150	\$1800
\$100	\$1200
\$130	\$1380
\$200	??

Now, the company wants to do the advertisement of \$200 in the year 2019 **and wants to know the prediction about the sales for this year**. So to solve such type of prediction problems in machine learning, we need regression analysis.

Regression is a supervised learning technique which helps in finding the correlation between variables and enables us to predict the continuous output variable based on the one or more predictor variables. It is mainly used for **prediction, forecasting, time series modeling, and determining the causal-effect relationship between variables**.

In Regression, we plot a graph between the variables which best fits the given datapoints, using this plot, the machine learning model can make predictions about the data. In simple words, ***"Regression shows a line or curve that passes through all the datapoints on target-predictor graph in such a way that the vertical distance between the datapoints and the regression line is minimum."*** The distance between datapoints and line tells whether a model has captured a strong relationship or not.

Some examples of regression can be as:

- Prediction of rain using temperature and other factors
- Determining Market trends
- Prediction of road accidents due to rash driving.

### **Terminologies Related to the Regression Analysis:**

- **Dependent Variable:** The main factor in Regression analysis which we want to predict or understand is called the dependent variable. It is also called **target variable**.
- **Independent Variable:** The factors which affect the dependent variables or which are used to predict the values of the dependent variables are called independent variable, also called as a **predictor**.
- **Outliers:** Outlier is an observation which contains either very low value or very high value in comparison to other observed values. An outlier may hamper the result, so it should be avoided.
- **Multicollinearity:** If the independent variables are highly correlated with each other than other variables, then such condition is called Multicollinearity. It should not be present in the dataset, because it creates problem while ranking the most affecting variable.
- **Underfitting and Overfitting:** If our algorithm works well with the training dataset but not well with test dataset, then such problem is called **Overfitting**. And if our algorithm does not perform well even with training dataset, then such problem is called **underfitting**.

## Why do we use Regression Analysis?

As mentioned above, Regression analysis helps in the prediction of a continuous variable. There are various scenarios in the real world where we need some future predictions such as weather condition, sales prediction, marketing trends, etc., for such case we need some technology which can make predictions more accurately. So for such case we need Regression analysis which is a statistical method and used in machine learning and data science. Below are some other reasons for using Regression analysis:

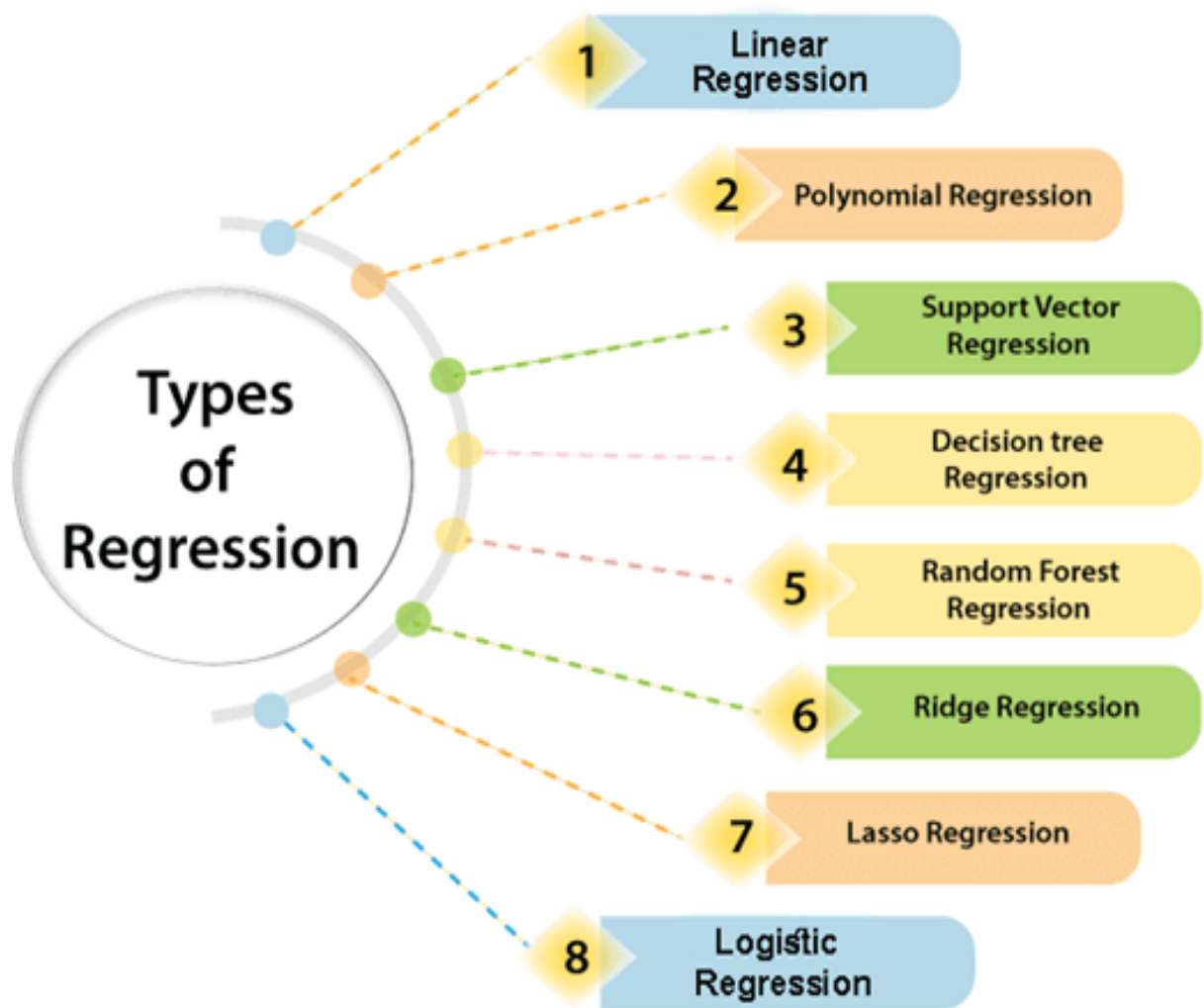
- Regression estimates the relationship between the target and the independent variable.
- It is used to find the trends in data.
- It helps to predict real/continuous values.
- By performing the regression, we can confidently determine the **most important factor, the least important factor, and how each factor is affecting the other factors.**

## Types of Regression

There are various types of regressions which are used in data science and machine learning. Each type has its own importance on different scenarios, but at the core, all the regression methods analyze the effect of the independent variable on dependent variables. Here we are discussing some important types of regression which are given below:

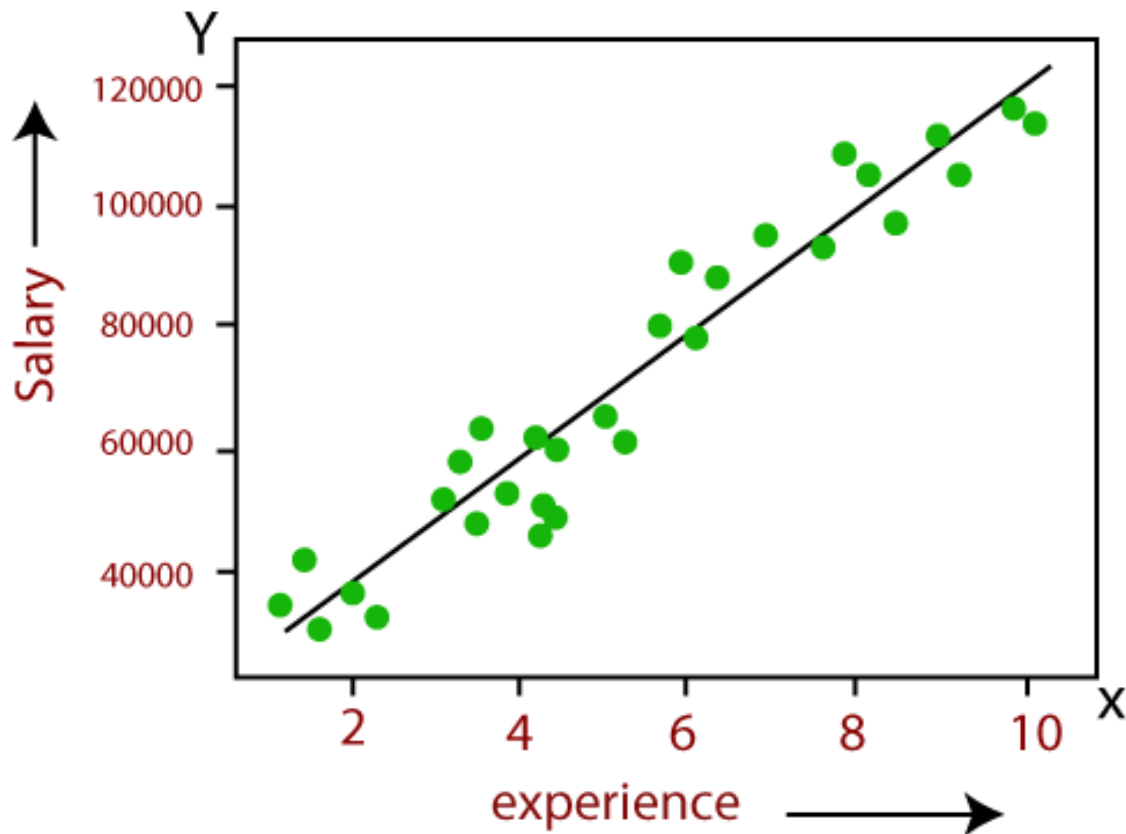
- **Linear Regression**
- **Logistic Regression**
- **Polynomial Regression**
- **Support Vector Regression**
- **Decision Tree Regression**
- **Random Forest Regression**
- **Ridge Regression**
- **Lasso Regression:**





## 1. Linear Regression:

- Linear regression is a statistical regression method which is used for predictive analysis.
- It is one of the very simple and easy algorithms which works on regression and shows the relationship between the continuous variables.
- It is used for solving the regression problem in machine learning.
- Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression.
- If there is only one input variable ( $x$ ), then such linear regression is called **simple linear regression**. And if there is more than one input variable, then such linear regression is called **multiple linear regression**.
- The relationship between variables in the linear regression model can be explained using the below image. Here we are predicting the salary of an employee on the basis of **the year of experience**.



- Below is the mathematical equation for Linear regression:

$$Y = aX + b$$

Here, Y = dependent variables (target variables),  
X = Independent variables (predictor variables),  
a and b are the linear coefficients

Some popular applications of linear regression are:

- Analyzing trends and sales estimates**
- Salary forecasting**
- Real estate prediction**
- Arriving at ETAs in traffic.**

## 2. Logistic Regression:

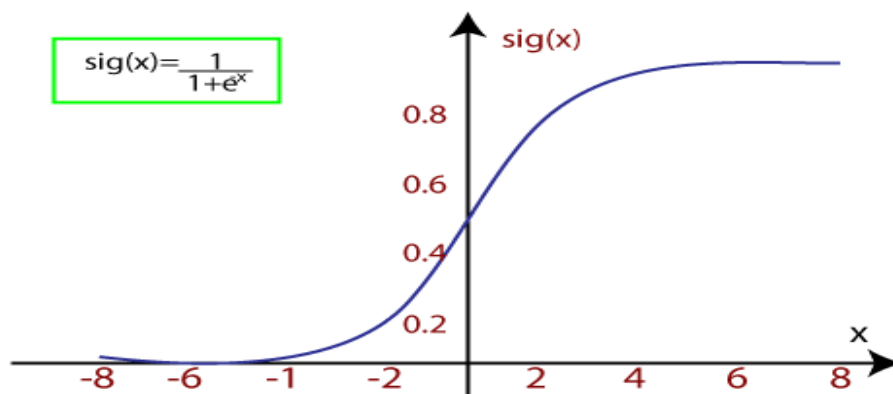
- Logistic regression is another supervised learning algorithm which is used to solve the classification problems. In **classification problems**, we have dependent variables in a binary or discrete format such as 0 or 1.
- Logistic regression algorithm works with the categorical variable such as 0 or 1, Yes or No, True or False, Spam or not spam, etc.

- It is a predictive analysis algorithm which works on the concept of probability.
- Logistic regression is a type of regression, but it is different from the linear regression algorithm in the term how they are used.
- Logistic regression uses **sigmoid function** or logistic function which is a complex cost function. This sigmoid function is used to model the data in logistic regression. The function can be represented as:

$$f(x) = \frac{1}{1+e^{-x}}$$

- $f(x)$  = Output between the 0 and 1 value.
- $x$  = input to the function
- $e$  = base of natural logarithm.

When we provide the input values (data) to the function, it gives the S-curve as follows:



- It uses the concept of threshold levels, values above the threshold level are rounded up to 1, and values below the threshold level are rounded up to 0.

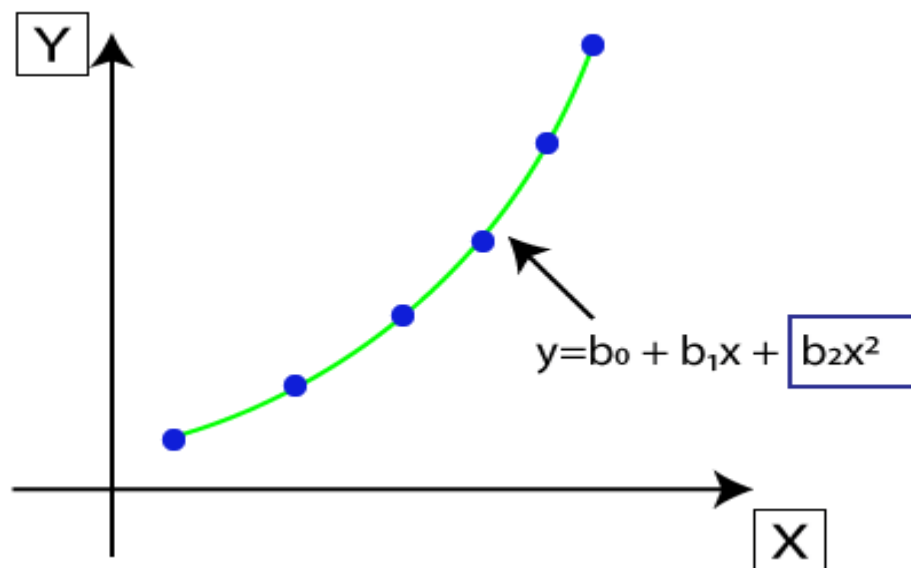
There are three types of logistic regression:

- **Binary(0/1, pass/fail)**
- **Multi(cats, dogs, lions)**
- **Ordinal(low, medium, high)**

### 3. Polynomial Regression:

- Polynomial Regression is a type of regression which models the **non-linear dataset** using a linear model.
- It is similar to multiple linear regression, but it fits a non-linear curve between the value of  $x$  and corresponding conditional values of  $y$ .

- Suppose there is a dataset which consists of datapoints which are present in a non-linear fashion, so for such case, linear regression will not best fit to those datapoints. To cover such datapoints, we need Polynomial regression.
- In **Polynomial regression**, the **original features are transformed into polynomial features of given degree and then modeled using a linear model**. Which means the datapoints are best fitted using a polynomial line.



- The equation for polynomial regression also derived from linear regression equation that means Linear regression equation  $Y = b_0 + b_1x$ , is transformed into Polynomial regression equation  $Y = b_0 + b_1x + b_2x^2 + b_3x^3 + \dots + b_nx^n$ .
- Here  $Y$  is the **predicted/target output**,  $b_0, b_1, \dots, b_n$  are the **regression coefficients**.  $x$  is our **independent/input variable**.
- The model is still linear as the coefficients are still linear with quadratic

**Note:** This is different from Multiple Linear regression in such a way that in Polynomial regression, a single element has different degrees instead of multiple variables with the same degree.

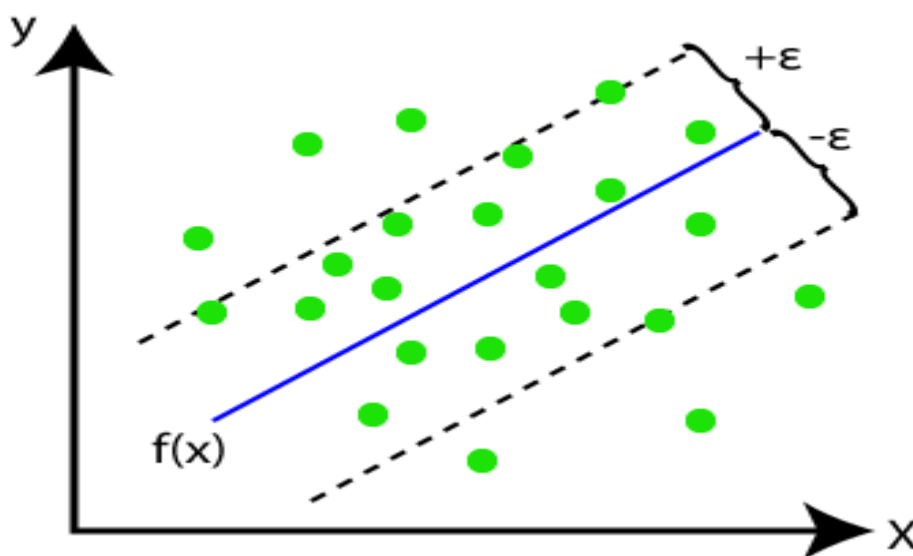
#### 4. Support Vector Regression:

Support Vector Machine is a supervised learning algorithm which can be used for regression as well as classification problems. So if we use it for regression problems, then it is termed as Support Vector Regression.

Support Vector Regression is a regression algorithm which works for continuous variables. Below are some keywords which are used in **Support Vector Regression**:

- **Kernel:** It is a function used to map a lower-dimensional data into higher dimensional data.
- **Hyperplane:** In general SVM, it is a separation line between two classes, but in SVR, it is a line which helps to predict the continuous variables and cover most of the datapoints.
- **Boundary line:** Boundary lines are the two lines apart from hyperplane, which creates a margin for datapoints.
- **Support vectors:** Support vectors are the datapoints which are nearest to the hyperplane and opposite class.

In SVR, we always try to determine a hyperplane with a maximum margin, so that maximum number of datapoints are covered in that margin. ***The main goal of SVR is to consider the maximum datapoints within the boundary lines and the hyperplane (best-fit line) must contain a maximum number of datapoints.*** Consider the below image:

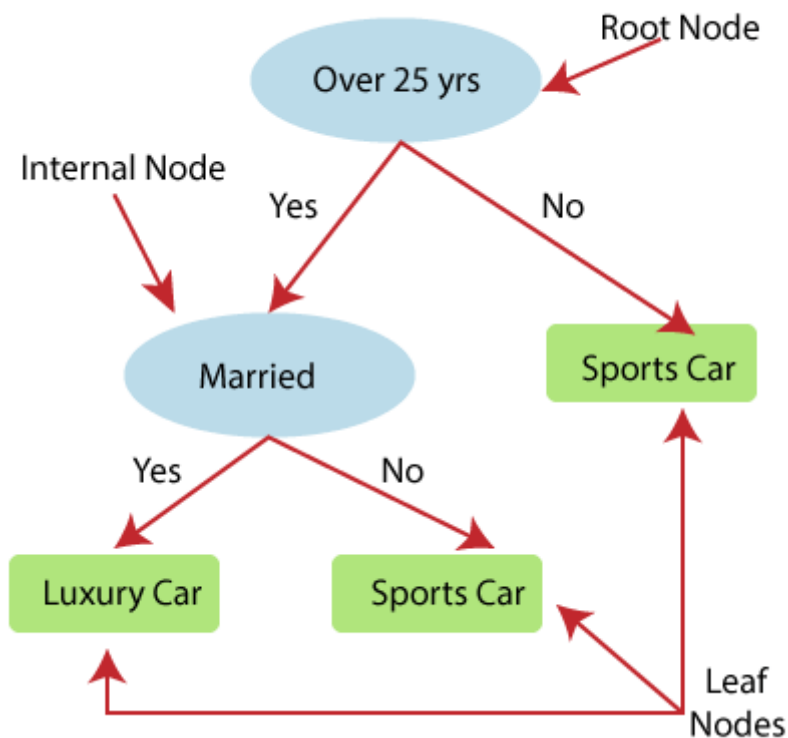


Here, the blue line is called hyperplane, and the other two lines are known as boundary lines.

## 5. Decision Tree Regression:

- Decision Tree is a supervised learning algorithm which can be used for solving both classification and regression problems.
- It can solve problems for both categorical and numerical data
- Decision Tree regression builds a tree-like structure in which each internal node represents the "test" for an attribute, each branch represent the result of the test, and each leaf node represents the final decision or result.

- A decision tree is constructed starting from the root node/parent node (dataset), which splits into left and right child nodes (subsets of dataset). These child nodes are further divided into their children node, and themselves become the parent node of those nodes. Consider the below image:

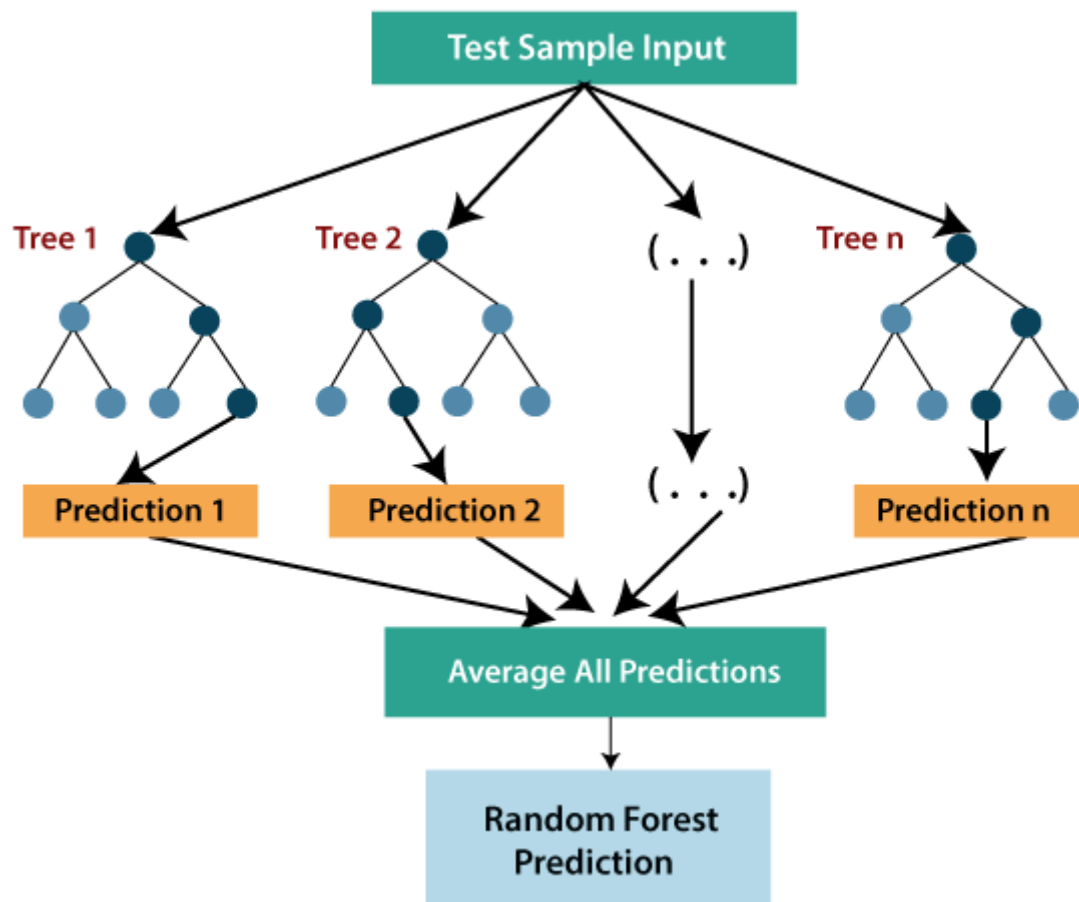


Above image showing the example of Decision Tree regression, here, the model is trying to predict the choice of a person between Sports cars or Luxury car.

- Random forest is one of the most powerful supervised learning algorithms which is capable of performing regression as well as classification tasks.
- The Random Forest regression is an ensemble learning method which combines multiple decision trees and predicts the final output based on the average of each tree output. The combined decision trees are called as base models, and it can be represented more formally as:

$$g(x) = f_0(x) + f_1(x) + f_2(x) + \dots$$

- Random forest uses **Bagging or Bootstrap Aggregation** technique of ensemble learning in which aggregated decision tree runs in parallel and do not interact with each other.
- With the help of Random Forest regression, we can prevent Overfitting in the model by creating random subsets of the dataset.



## 6. Ridge Regression:

- Ridge regression is one of the most robust versions of linear regression in which a small amount of bias is introduced so that we can get better long term predictions.
- The amount of bias added to the model is known as **Ridge Regression penalty**. We can compute this penalty term by multiplying with the lambda to the squared weight of each individual features.
- The equation for ridge regression will be:

$$L(x, y) = \text{Min}(\sum_{i=1}^n (y_i - w_i x_i)^2 + \lambda \sum_{i=1}^n (w_i)^2)$$

- A general linear or polynomial regression will fail if there is high collinearity between the independent variables, so to solve such problems, Ridge regression can be used.
- Ridge regression is a regularization technique, which is used to reduce the complexity of the model. It is also called as **L2 regularization**.
- It helps to solve the problems if we have more parameters than samples.

## 7. Lasso Regression:

- Lasso regression is another regularization technique to reduce the complexity of the model.
- It is similar to the Ridge Regression except that penalty term contains only the absolute weights instead of a square of weights.
- Since it takes absolute values, hence, it can shrink the slope to 0, whereas Ridge Regression can only shrink it near to 0.
- It is also called as **L1 regularization**. The equation for Lasso regression will be:

$$L(x, y) = \text{Min} \left( \sum_{i=1}^n (y_i - w_i x_i)^2 + \lambda \sum_{i=1}^n |w_i| \right)$$

### Classification algorithms:

As we know, the Supervised Machine Learning algorithm can be broadly classified into Regression and Classification Algorithms. In Regression algorithms, we have predicted the output for continuous values, but to predict the categorical values, we need Classification algorithms.

### What is the Classification Algorithm?

The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data. In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, **Yes or No, 0 or 1, Spam or Not Spam, cat or dog**, etc. Classes can be called as targets/labels or categories.

Unlike regression, the output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc. Since the Classification algorithm is a Supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output.

In classification algorithm, a discrete output function( $y$ ) is mapped to input variable( $x$ ).

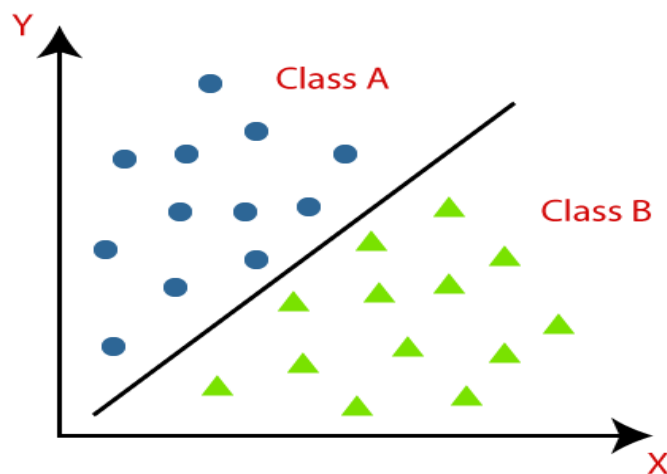
$$y=f(x), \text{ where } y = \text{categorical output}$$

The best example of an ML classification algorithm is **Email Spam Detector**.

The main goal of the Classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data.



Classification algorithms can be better understood using the below diagram. In the below diagram, there are two classes, class A and Class B. These classes have features that are similar to each other and dissimilar to other classes.



The algorithm which implements the classification on a dataset is known as a classifier. There are two types of Classifications:

- **Binary Classifier:** If the classification problem has only two possible outcomes, then it is called as Binary Classifier. **Examples:** YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.
- **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called as Multi-class Classifier. **Example:** Classifications of types of crops, Classification of types of music.

### Learners in Classification Problems:

In the classification problems, there are two types of learners:

1. **Lazy Learners:** Lazy Learner firstly stores the training dataset and wait until it receives the test dataset. In Lazy learner case, classification is done on the basis of the most related data stored in the training dataset. It takes less time in training but more time for predictions. **Example:** K-NN algorithm, Case-based reasoning
2. **Eager Learners:** Eager Learners develop a classification model based on a training dataset before receiving a test dataset. Opposite to Lazy learners, Eager Learner takes more time in learning, and less time in prediction. **Example:** Decision Trees, Naïve Bayes, ANN.

## Types of ML Classification Algorithms:

Classification Algorithms can be further divided into the Mainly two category:

- **Linear Models**
  - Logistic Regression
  - Support Vector Machines
- **Non-linear Models**
  - K-Nearest Neighbours
  - Kernel SVM
  - Naïve Bayes
  - Decision Tree Classification
  - Random Forest Classification

Note: We will learn the above algorithms in later chapters.

## Evaluating a Classification model:

Once our model is completed, it is necessary to evaluate its performance; either it is a Classification or Regression model. So for evaluating a Classification model, we have the following ways:

### 1. Log Loss or Cross-Entropy Loss:

- It is used for evaluating the performance of a classifier, whose output is a probability value between the 0 and 1.
- For a good binary Classification model, the value of log loss should be near to 0.
- The value of log loss increases if the predicted value deviates from the actual value.
- The lower log loss represents the higher accuracy of the model.
- For Binary classification, cross-entropy can be calculated as:

$$1. -(y \log(p) + (1-y) \log(1-p))$$

Where  $y$  = Actual output,  $p$  = predicted output.

### 2. Confusion Matrix:

- The confusion matrix provides us a matrix/table as output and describes the performance of the model.
- It is also known as the error matrix.

- The matrix consists of predictions result in a summarized form, which has a total number of correct predictions and incorrect predictions. The matrix looks like as below table:

	Actual Positive	Actual Negative
Predicted Positive	True Positive	False Positive
Predicted Negative	False Negative	True Negative

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total Population}}$$

### 3. AUC-ROC curve:

- ROC curve stands for **Receiver Operating Characteristics Curve** and AUC stands for **Area Under the Curve**.
- It is a graph that shows the performance of the classification model at different thresholds.
- To visualize the performance of the multi-class classification model, we use the AUC-ROC Curve.
- The ROC curve is plotted with TPR and FPR, where TPR (True Positive Rate) on Y-axis and FPR(False Positive Rate) on X-axis.

### Use cases of Classification Algorithms

Classification algorithms can be used in different places. Below are some popular use cases of Classification Algorithms:

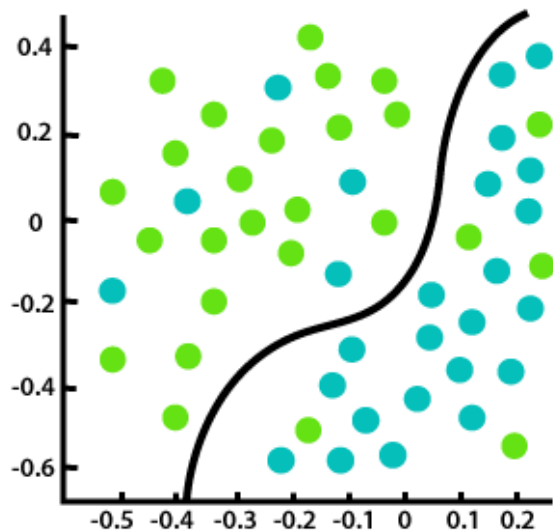
- Email Spam Detection
- Speech Recognition
- Identifications of Cancer tumor cells.
- Drugs Classification
- Biometric Identification, etc.

## REGRESSION VS. CLASSIFICATION

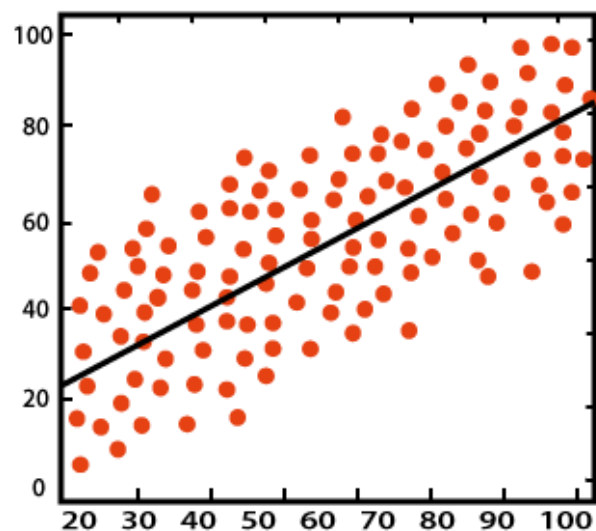
Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labelled datasets. But the difference between both is how they are used for different machine learning problems.

The main difference between Regression and Classification algorithms is that Regression algorithms are used to **predict the continuous** values such as price, salary, age, etc. and Classification algorithms are used to **predict/Classify the discrete values** such as Male or Female, True or False, Spam or Not Spam, etc.

Consider the below diagram:



Classification



Regression

### Classification:

Classification is a process of finding a function which helps in dividing the dataset into classes based on different parameters. In Classification, a computer program is trained on the training dataset and based on that training, it categorizes the data into different classes.

The task of the classification algorithm is to find the mapping function to map the input( $x$ ) to the discrete output( $y$ ).

**Example:** The best example to understand the Classification problem is Email Spam Detection. The model is trained on the basis of millions of emails on different parameters, and whenever it receives a new email, it identifies whether the email is spam or not. If the email is spam, then it is moved to the Spam folder.

## Types of ML Classification Algorithms:

Classification Algorithms can be further divided into the following types:

- Logistic Regression
- K-Nearest Neighbours
- Support Vector Machines
- Kernel SVM
- Naïve Bayes
- Decision Tree Classification
- Random Forest Classification

## Regression:

Regression is a process of finding the correlations between dependent and independent variables. It helps in predicting the continuous variables such as prediction of **Market Trends**, prediction of House prices, etc.

The task of the Regression algorithm is to find the mapping function to map the input variable(x) to the continuous output variable(y).

**Example:** Suppose we want to do weather forecasting, so for this, we will use the Regression algorithm. In weather prediction, the model is trained on the past data, and once the training is completed, it can easily predict the weather for future days.

## Types of Regression Algorithm:

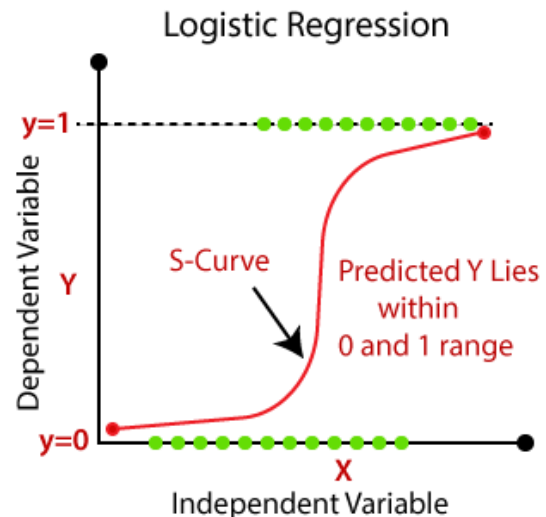
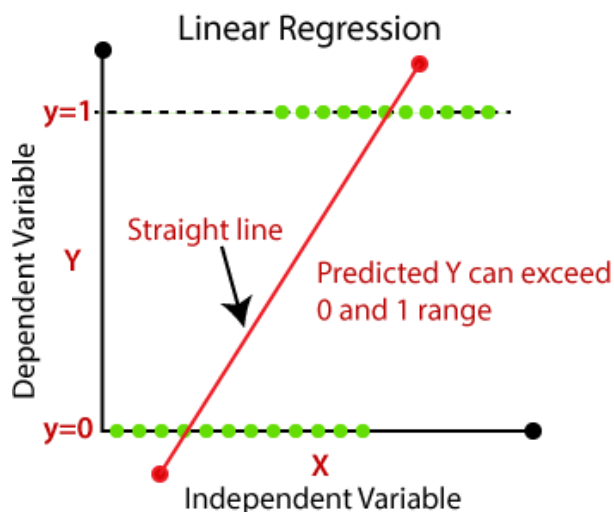
- Simple Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Support Vector Regression
- Decision Tree Regression
- Random Forest Regression

## Difference between Regression and Classification

Regression Algorithm	Classification Algorithm
In Regression, the output variable must be of continuous nature or real value.	In Classification, the output variable must be a discrete value.
The task of the regression algorithm is to map the input value (x) with the continuous output variable(y).	The task of the classification algorithm is to map the input value(x) with the discrete output variable(y).
Regression Algorithms are used with continuous data.	Classification Algorithms are used with discrete data.
In Regression, we try to find the best fit line, which can predict the output more accurately.	In Classification, we try to find the decision boundary, which can divide the dataset into different classes.
Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, etc.	Classification Algorithms can be used to solve classification problems such as Identification of spam emails, Speech Recognition, Identification of cancer cells, etc.
The regression Algorithm can be further divided into Linear and Non-linear Regression.	The Classification algorithms can be divided into Binary Classifier and Multi-class Classifier.

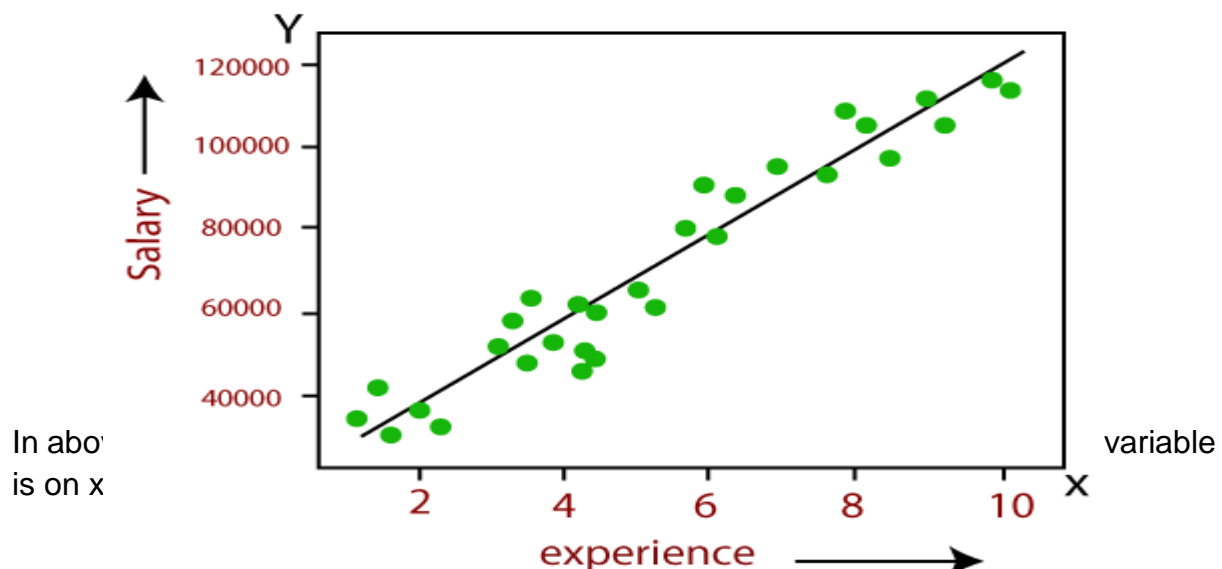
## Linear Regression vs Logistic Regression

Linear Regression and Logistic Regression are the two famous Machine Learning Algorithms which come under supervised learning technique. Since both the algorithms are of supervised in nature hence these algorithms use labelled dataset to make the predictions. But the main difference between them is how they are being used. The Linear Regression is used for solving Regression problems whereas Logistic Regression is used for solving the Classification problems. The description of both the algorithms is given below along with difference table.



### Linear Regression:

- Linear Regression is one of the most simple Machine learning algorithm that comes under Supervised Learning technique and used for solving regression problems.
- It is used for predicting the continuous dependent variable with the help of independent variables.
- The goal of the Linear regression is to find the best fit line that can accurately predict the output for the continuous dependent variable.
- If single independent variable is used for prediction then it is called Simple Linear Regression and if there are more than two independent variables then such regression is called as Multiple Linear Regression.
- By finding the best fit line, algorithm establish the relationship between dependent variable and independent variable. And the relationship should be of linear nature.
- The output for Linear regression should only be the continuous values such as price, age, salary, etc. The relationship between the dependent variable and independent variable can be shown in below image:

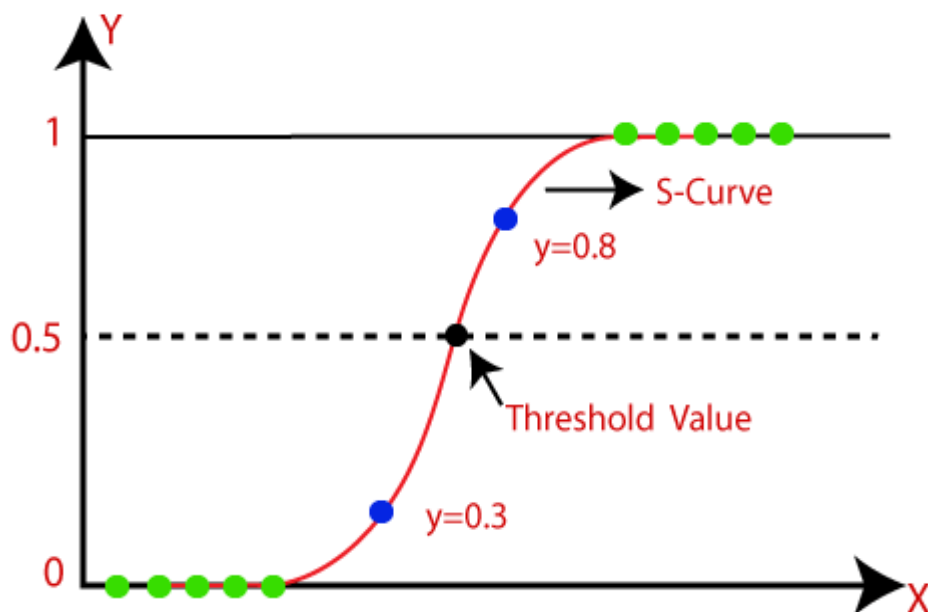


$$y = a_0 + a_1x + \varepsilon$$

Where,  $a_0$  and  $a_1$  are the coefficients and  $\varepsilon$  is the error term.

## Logistic Regression:

- Logistic regression is one of the most popular Machine learning algorithm that comes under Supervised Learning techniques.
- It can be used for Classification as well as for Regression problems, but mainly used for Classification problems.
- Logistic regression is used to predict the categorical dependent variable with the help of independent variables.
- The output of Logistic Regression problem can be only between the 0 and 1.
- Logistic regression can be used where the probabilities between two classes is required. Such as whether it will rain today or not, either 0 or 1, true or false etc.
- Logistic regression is based on the concept of Maximum Likelihood estimation. According to this estimation, the observed data should be most probable.
- In logistic regression, we pass the weighted sum of inputs through an activation function that can map values in between 0 and 1. Such activation function is known as **sigmoid function** and the curve obtained is called as sigmoid curve or S-curve. Consider the below image:



- The equation for logistic regression is:

$$\log \left[ \frac{y}{1-y} \right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$



## Difference between Linear Regression and Logistic Regression:

Linear Regression	Logistic Regression
Linear regression is used to predict the continuous dependent variable using a given set of independent variables.	Logistic Regression is used to predict the categorical dependent variable using a given set of independent variables.
Linear Regression is used for solving Regression problem.	Logistic regression is used for solving Classification problems.
In Linear regression, we predict the value of continuous variables.	In logistic Regression, we predict the values of categorical variables.
In linear regression, we find the best fit line, by which we can easily predict the output.	In Logistic Regression, we find the S-curve by which we can classify the samples.
Least square estimation method is used for estimation of accuracy.	Maximum likelihood estimation method is used for estimation of accuracy.
The output for Linear Regression must be a continuous value, such as price, age, etc.	The output of Logistic Regression must be a Categorical value such as 0 or 1, Yes or No, etc.
In Linear regression, it is required that relationship between dependent variable and independent variable must be linear.	In Logistic regression, it is not required to have the linear relationship between the dependent and independent variable.
In linear regression, there may be collinearity between the independent variables.	In logistic regression, there should not be collinearity between the independent variable.

## SEMI-SUPERVISED LEARNING

*Semi-Supervised learning is a type of Machine Learning algorithm that represents the intermediate ground between Supervised and Unsupervised learning algorithms. It uses the combination of labeled and unlabeled datasets during the training period.*



Before understanding the Semi-Supervised learning, you should know the main categories of Machine Learning algorithms. Machine Learning consists of three main categories: **Supervised Learning**, **Unsupervised Learning**, and **Reinforcement Learning**. Further, the basic difference between Supervised and unsupervised learning is that *supervised learning datasets consist of an output label training data associated with each tuple*, and *unsupervised datasets do not consist the same*. ***Semi-supervised learning is an important category that lies between the Supervised and Unsupervised machine learning.*** Although Semi-supervised learning is the middle ground between supervised and unsupervised learning and operates on the data that consists of a few labels, it mostly consists of unlabeled data. As labels are costly, but for the corporate purpose, it may have few labels.

The basic disadvantage of supervised learning is that it requires hand-labeling by ML specialists or data scientists, and it also requires a high cost to process. Further unsupervised learning also has a limited spectrum for its applications. **To overcome these drawbacks of supervised learning and unsupervised learning algorithms, the concept of Semi-supervised learning is introduced.** In this algorithm, training data is a combination of both labeled and unlabeled data. However, labeled data exists with a very small amount while it consists of a huge amount of unlabeled data. Initially, similar data is clustered along with an unsupervised learning algorithm, and further, it helps to label the unlabeled data into labeled data. It is why label data is a comparatively, more expensive acquisition than unlabeled data.

We can imagine these algorithms with an example. Supervised learning is where a student is under the supervision of an instructor at home and college. Further, if that student is self-analyzing the same concept without any help from the instructor, it comes under unsupervised learning. Under semi-supervised learning, the student has to revise itself after analyzing the same concept under the guidance of an instructor at college.

## Assumptions followed by Semi-Supervised Learning

To work with the unlabeled dataset, there must be a relationship between the objects. To understand this, semi-supervised learning uses any of the following assumptions:

- **Continuity Assumption:**  
As per the continuity assumption, the objects near each other tend to share the same group or label. This assumption is also used in supervised learning, and the datasets are separated by the decision boundaries. But in semi-supervised, the decision boundaries are added with the smoothness assumption in low-density boundaries.
- **Cluster assumptions-** In this assumption, data are divided into different discrete clusters. Further, the points in the same cluster share the output label.
- **Manifold assumptions-** This assumption helps to use distances and densities, and this data lie on a manifold of fewer dimensions than input space.
- The dimensional data are created by a process that has less degree of freedom and may be hard to model directly. **(This assumption becomes practical if high).**

## Working of Semi-Supervised Learning

Semi-supervised learning uses pseudo labeling to train the model with less labeled training data than supervised learning. The process can combine various neural network models and training ways. The whole working of semi-supervised learning is explained in the below points:

- Firstly, it trains the model with less amount of training data similar to the supervised learning models. The training continues until the model gives accurate results.
- The algorithms use the unlabeled dataset with pseudo labels in the next step, and now the result may not be accurate.
- Now, the labels from labeled training data and pseudo labels data are linked together.
- The input data in labeled training data and unlabeled training data are also linked.

- In the end, again train the model with the new combined input as did in the first step. It will reduce errors and improve the accuracy of the model.

## **Difference between Semi-supervised and Reinforcement Learning.**

Reinforcement learning is different from semi-supervised learning, as it works with rewards and feedback. ***Reinforcement learning aims to maximize the rewards by their hit and trial actions, whereas in semi-supervised learning, we train the model with a less labeled dataset.***

## **Real-world applications of Semi-supervised Learning-**

Semi-supervised learning models are becoming more popular in the industries. Some of the main applications are as follows.

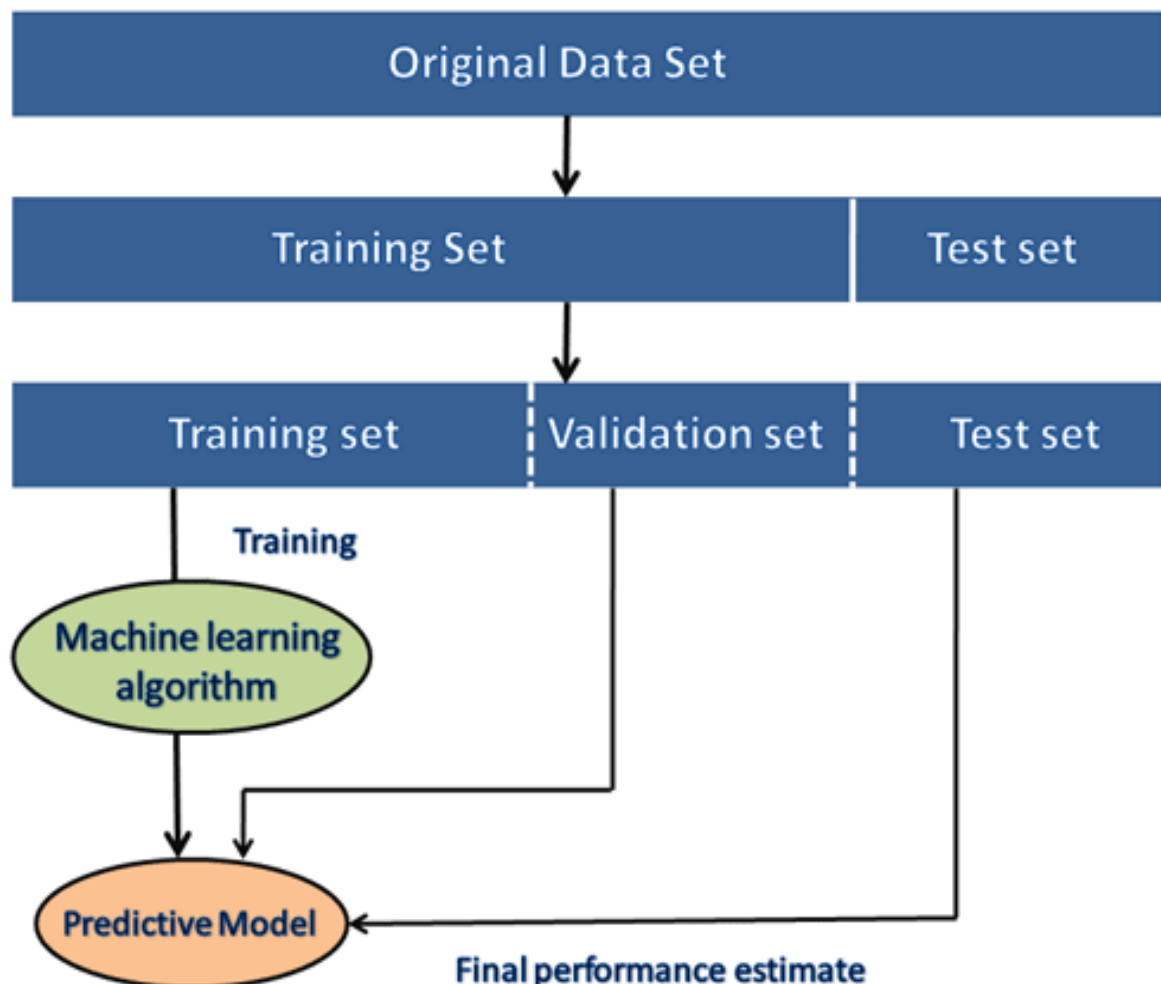
- **Speech Analysis-** It is the most classic example of semi-supervised learning applications. Since, labeling the audio data is the most impassable task that requires many human resources, this problem can be naturally overcome with the help of applying SSL in a Semi-supervised learning model.
- **Web content classification-** However, this is very critical and impossible to label each page on the internet because it needs mode human intervention. Still, this problem can be reduced through Semi-Supervised learning algorithms.  
Further, Google also uses semi-supervised learning algorithms to rank a webpage for a given query.
- **Protein sequence classification-** DNA strands are larger, they require active human intervention. So, the rise of the Semi-supervised model has been proximate in this field.
- **Text document classifier-** As we know, it would be very unfeasible to find a large amount of labeled text data, so semi-supervised learning is an ideal model to overcome this

# DATA PRE-PROCESSING

Data pre-processing is a fundamental stage in preparing datasets for machine learning. It includes changing raw data into a configuration reasonable for model training. Normal pre-processing procedures incorporate data cleaning to eliminate irregularities or blunders, standardization to scale data inside a particular reach, highlight scaling to guarantee highlights have comparative ranges, and taking care of missing qualities through ascription or evacuation.

During the development of the ML project, the developers completely rely on the datasets. In building ML applications, datasets are divided into two parts:

- **Training dataset**
- **Test Dataset**



Note: The datasets are of large size, so to download these datasets, you must have fast internet on your computer.

## **Training Dataset and Test Dataset:**

In machine learning, datasets are ordinarily partitioned into two sections: the training dataset and the test dataset. The training dataset is utilized to prepare the machine learning model, while the test dataset is utilized to assess the model's exhibition. This division surveys the model's capacity, to sum up to inconspicuous data. It is fundamental to guarantee that the datasets are representative of the issue space and appropriately split to stay away from inclination or overfitting.

## **Popular sources for Machine Learning datasets**

Below is the list of datasets which are freely available for the public to work on it:

### **1. Kaggle Datasets**

Kaggle is one of the best sources for providing datasets for Data Scientists and Machine Learners. It allows users to find, download, and publish datasets in an easy way. It also provides the opportunity to work with other machine learning engineers and solve difficult Data Science related tasks.

Kaggle provides a high-quality dataset in different formats that we can easily find and download.

The link for the Kaggle dataset is <https://www.kaggle.com/datasets>.

### **2. UCI Machine Learning Repository**

The UCI Machine Learning Repository is an important asset that has been broadly utilized by scientists and specialists beginning around 1987. It contains a huge collection of datasets sorted by machine learning tasks such as regression, classification, and clustering. Remarkable datasets in the storehouse incorporate the Iris dataset, Vehicle Assessment dataset, and Poker Hand dataset.

The link for the UCI machine learning repository is <https://archive.ics.uci.edu/ml/index.php>.

### **3. Datasets via AWS**

We can search, download, access, and share the datasets that are publicly available via AWS resources. These datasets can be accessed through AWS resources but

provided and maintained by different government organizations, researches, businesses, or individuals.

Anyone can analyze and build various services using shared data via AWS resources. The shared dataset on cloud helps users to spend more time on data analysis rather than on acquisitions of data.

This source provides the various types of datasets with examples and ways to use the dataset. It also provides the search box using which we can search for the required dataset. Anyone can add any dataset or example to the **Registry of Open Data on AWS**.

The link for the resource is <https://registry.opendata.aws/>.

#### 4. Google's Dataset Search Engine

Google's Dataset Web index helps scientists find and access important datasets from different sources across the web. It files datasets from areas like sociologies, science, and environmental science. Specialists can utilize catchphrases to find datasets, channel results in light of explicit standards, and access the datasets straightforwardly from the source.

The link for the Google dataset search engine is <https://toolbox.google.com/datasetsearch>.

#### 5. Microsoft Datasets

The Microsoft has launched the "**Microsoft Research Open data**" repository with the collection of free datasets in various areas such as **natural language processing, computer vision, and domain-specific sciences**. It gives admittance to assorted and arranged datasets that can be significant for machine learning projects.

The link to download or use the dataset from this resource is <https://msropendata.com/>.

#### 6. Awesome Public Dataset Collection

Awesome public dataset collection provides high-quality datasets that are arranged in a well-organized manner within a list according to topics such as Agriculture, Biology, Climate, Complex networks, etc. Most of the datasets are available free, but some may not, so it is better to check the license before downloading the dataset.

The link to download the dataset from Awesome public dataset collection is <https://github.com/awesomedata/awesome-public-datasets>.

## 7. Government Datasets

There are different sources to get government-related data. Various countries publish government data for public use collected by them from different departments.

The goal of providing these datasets is to increase transparency of government work among the people and to use the data in an innovative approach. Below are some links of government datasets:

- [Indian Government dataset](#)
- [US Government Dataset](#)
- [Northern Ireland Public Sector Datasets](#)
- [European Union Open Data Portal](#)

## 8. Computer Vision Datasets

Visual data provides multiple numbers of the great dataset that are specific to computer visions such as Image Classification, Video classification, Image Segmentation, etc. Therefore, if you want to build a project on deep learning or image processing, then you can refer to this source.

The link for downloading the dataset from this source is <https://www.visualdata.io/>.

## 9. Scikit-learn dataset

Scikit-learn, a well-known machine learning library in Python, gives a few underlying datasets to practice and trial and error. These datasets are open through the sci-kit-learn Programming interface and can be utilized for learning different machine-learning calculations. Scikit-learn offers both toy datasets, which are little and improved, and genuine world datasets with greater intricacy. Instances of sci-kit-learn datasets incorporate the Iris dataset, the Boston Lodging dataset, and the Wine dataset.

The link to download datasets from this source is <https://scikit-learn.org/stable/datasets/index.html>.

## Data Ethics and Privacy:

Data ethics and privacy are basic contemplations in machine learning projects. It is fundamental to guarantee that data is gathered and utilized morally, regarding privacy freedoms and observing pertinent regulations and guidelines. Data experts ought to go to lengths to safeguard data privacy, get appropriate assent, and handle delicate data mindfully. Assets, for example, moral rules and privacy structures can give direction on keeping up with moral practices in data assortment and use.



## **Conclusion:**

In conclusion, datasets structure the groundwork of effective machine-learning projects. Understanding the various kinds of datasets, the significance of data pre-processing, and the job of training and testing datasets are key stages towards building powerful models. By utilizing well-known sources, for example, Kaggle, UCI Machine Learning Repository, AWS, Google's Dataset Search, Microsoft Datasets, and government datasets, data researchers and specialists can get to an extensive variety of datasets for their machine learning projects. It is fundamental to consider data ethics and privacy all through the whole data lifecycle to guarantee mindful and moral utilization of data. With the right datasets and moral practices, machine learning models can accomplish exact predictions and drive significant bits of knowledge.

## **Data Preprocessing in Machine learning:**

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

## **Why do we need Data Preprocessing?**

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

- **Getting the dataset**
- **Importing libraries**
- **Importing datasets**
- **Finding Missing Data**
- **Encoding Categorical Data**
- **Splitting dataset into training and test set**
- **Feature scaling**

## 1) Get the Dataset

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the **dataset**.

Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. So each dataset is different from another dataset. To use the dataset in our code, we usually put it into a CSV **file**. However, sometimes, we may also need to use an HTML or xlsx file.

### What is a CSV File?

CSV stands for "**Comma-Separated Values**" files; it is a file format which allows us to save the tabular data, such as spreadsheets. It is useful for huge datasets and can use these datasets in programs.

Here we will use a demo dataset for data preprocessing, and for practice, it can be downloaded from here, "<https://www.superdatascience.com/pages/machine-learning>". For real-world problems, we can download datasets online from various sources such as <https://www.kaggle.com/uciml/datasets>, <https://archive.ics.uci.edu/ml/index.php> etc.

We can also create our dataset by gathering data using various API with Python and put that data into a .csv file.

## 2) Importing Libraries

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

**Numpy:** Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as:

1. `import numpy as nm`

Here we have used **nm**, which is a short name for Numpy, and it will be used in the whole program.

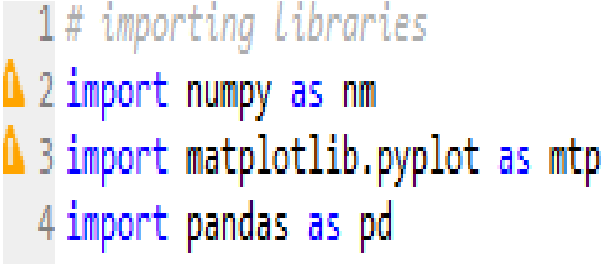
**Matplotlib:** The second library is **matplotlib**, which is a Python 2D plotting library, and with this library, we need to import a sub-library **pyplot**. This library is used to plot any type of charts in Python for the code. It will be imported as below:

1. `import matplotlib.pyplot as mpt`

Here we have used mpt as a short name for this library.

**Pandas:** The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. It will be imported as below:

Here, we have used pd as a short name for this library. Consider the below image:



```
1 # importing Libraries
2 import numpy as nm
3 import matplotlib.pyplot as mtp
4 import pandas as pd
```

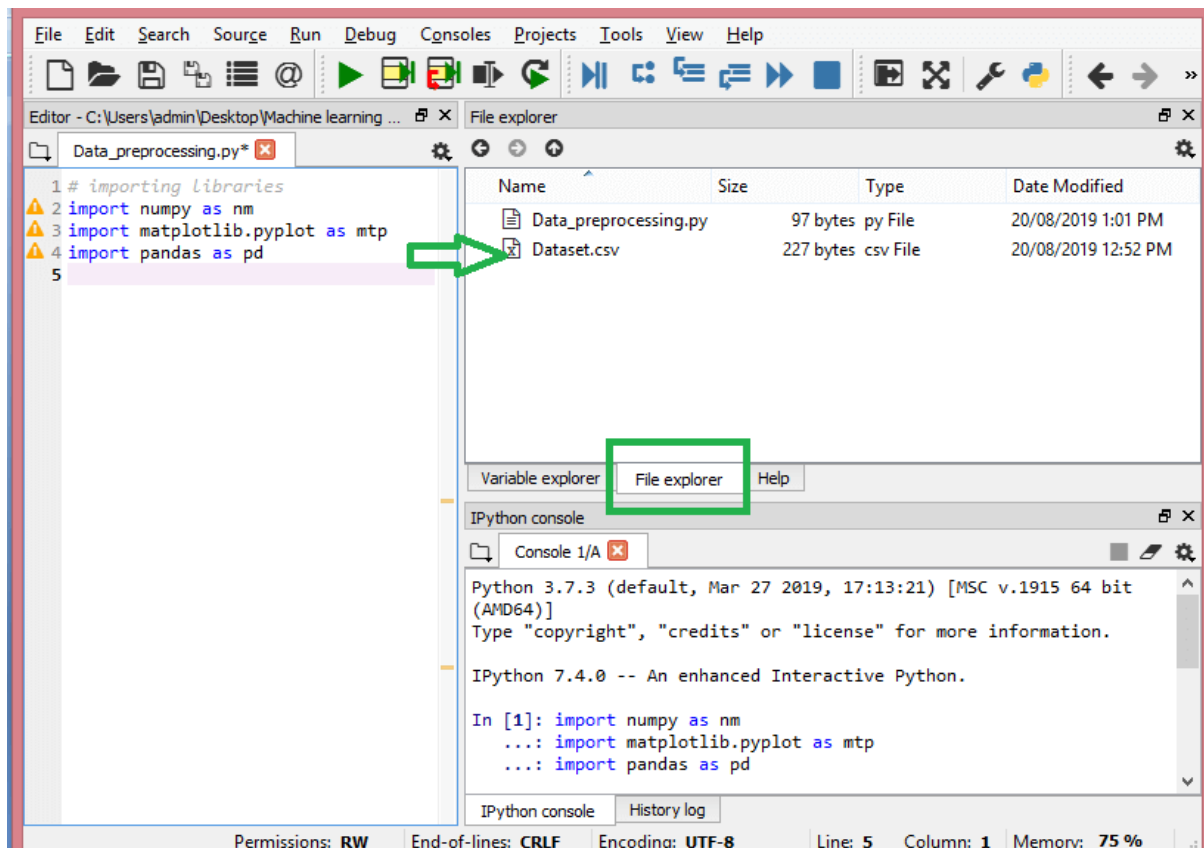
### 3) Importing the Datasets

Now we need to import the datasets which we have collected for our machine learning project. But before importing a dataset, we need to set the current directory as a working directory. To set a working directory in Spyder IDE, we need to follow the below steps:

1. Save your Python file in the directory which contains dataset.
2. Go to File explorer option in Spyder IDE, and select the required directory.
3. Click on F5 button or run option to execute the file.

**Note:** We can set any directory as a working directory, but it must contain the required dataset.

Here, in the below image, we can see the Python file along with required dataset. Now, the current folder is set as a working directory.



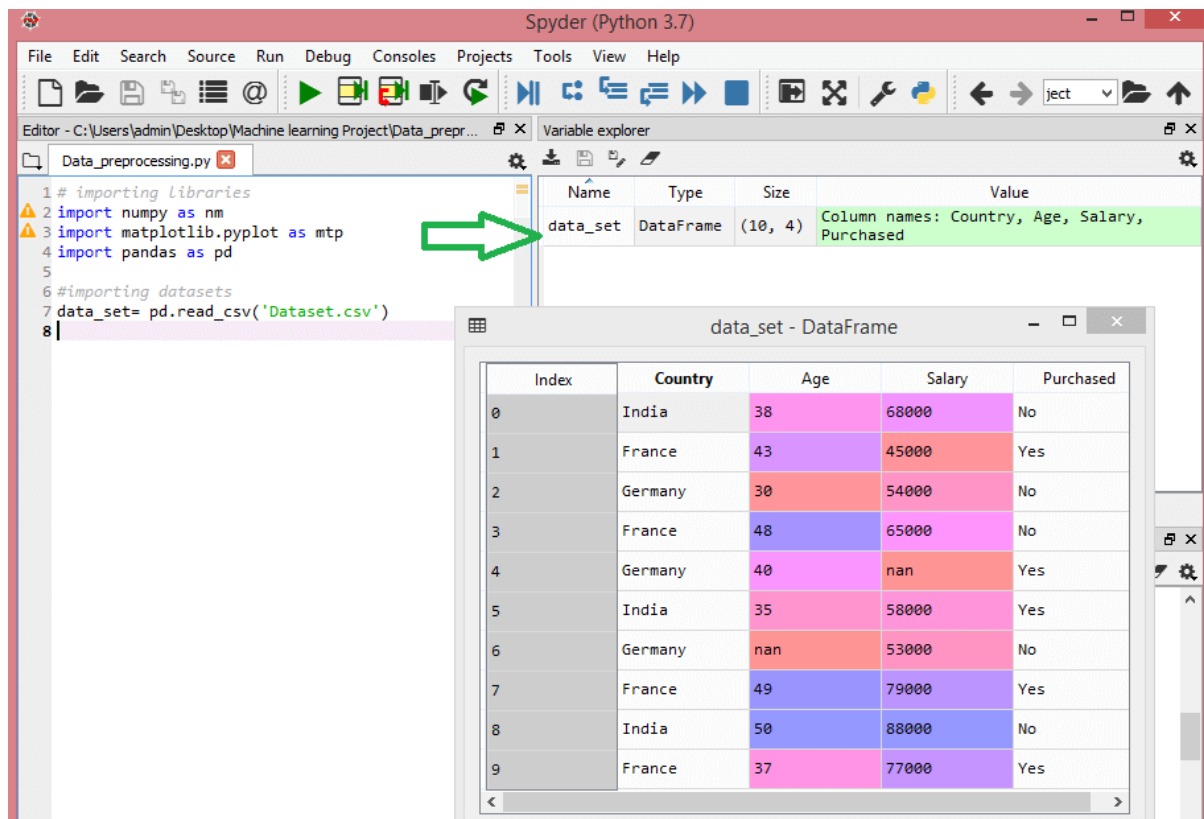
### read\_csv() function:

Now to import the dataset, we will use read\_csv() function of pandas library, which is used to read a csv file and performs various operations on it. Using this function, we can read a csv file locally as well as through an URL.

We can use read\_csv function as below:

1. data\_set= pd.read\_csv('Dataset.csv')

Here, **data\_set** is a name of the variable to store our dataset, and inside the function, we have passed the name of our dataset. Once we execute the above line of code, it will successfully import the dataset in our code. We can also check the imported dataset by clicking on the section **variable explorer**, and then double click on **data\_set**. Consider the below image:



As in the above image, indexing is started from 0, which is the default indexing in Python. We can also change the format of our dataset by clicking on the format option.

### Extracting dependent and independent variables:

In machine learning, it is important to distinguish the matrix of features (independent variables) and dependent variables from dataset. In our dataset, there are three independent variables that are **Country**, **Age**, and **Salary**, and one is a dependent variable which is **Purchased**.

### Extracting independent variable:

To extract an independent variable, we will use **iloc[ ]** method of Pandas library. It is used to extract the required rows and columns from the dataset.

1. `x = data_set.iloc[:, :-1].values`

In the above code, the first colon(:) is used to take all the rows, and the second colon(:) is for all the columns. Here we have used :-1, because we don't want to take the last column as it contains the dependent variable. So by doing this, we will get the matrix of features.

By executing the above code, we will get output as:

1. [['India' 38.0 68000.0]
2. ['France' 43.0 45000.0]
3. ['Germany' 30.0 54000.0]
4. ['France' 48.0 65000.0]
5. ['Germany' 40.0 nan]
6. ['India' 35.0 58000.0]
7. ['Germany' nan 53000.0]
8. ['France' 49.0 79000.0]
9. ['India' 50.0 88000.0]
10. ['France' 37.0 77000.0]]

As we can see in the above output, there are only three variables.

### Extracting dependent variable:

To extract dependent variables, again, we will use Pandas `.iloc[]` method.

1. `y= data_set.iloc[:,3].values`

Here we have taken all the rows with the last column only. It will give the array of dependent variables.

By executing the above code, we will get output as:

### Output:

```
array(['No', 'Yes', 'No', 'No', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes'],  
      dtype=object)
```

**Note:** If you are using Python language for machine learning, then extraction is mandatory, but for R language it is not required.

## 4) Handling Missing data:

The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

### Ways to handle missing data:

There are mainly two ways to handle missing data, which are:

**By deleting the particular row:** The first way is used to commonly deal with null values. In this way, we just delete the specific row or column which consists of null

values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.

**By calculating the mean:** In this way, we will calculate the mean of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc. Here, we will use this approach.

To handle missing values, we will use **Scikit-learn** library in our code, which contains various libraries for building machine learning models. Here we will use **Imputer** class of **sklearn.preprocessing** library. Below is the code for it:

1. #handling missing data (Replacing missing data with the mean value)
2. from sklearn.preprocessing import Imputer
3. imputer= Imputer(missing\_values = 'NaN', strategy='mean', axis = 0)
4. #Fitting imputer object to the independent variables x.
5. imputer.fit(x[:, 1:3])
6. #Replacing missing data with the calculated mean value
7. x[:, 1:3]= imputer.transform(x[:, 1:3])

#### Output:

```
array([[ 'India', 38.0, 68000.0],
       [ 'France', 43.0, 45000.0],
       [ 'Germany', 30.0, 54000.0],
       [ 'France', 48.0, 65000.0],
       [ 'Germany', 40.0, 65222.22222222222],
       [ 'India', 35.0, 58000.0],
       [ 'Germany', 41.111111111111114, 53000.0],
       [ 'France', 49.0, 79000.0],
       [ 'India', 50.0, 88000.0],
       [ 'France', 37.0, 77000.0]], dtype=object)
```

As we can see in the above output, the missing values have been replaced with the means of rest column values.

## 5) Encoding Categorical data:

Categorical data is data which has some categories such as, in our dataset; there are two categorical variable, **Country**, and **Purchased**.

Since machine learning model completely works on mathematics and numbers, but if our dataset would have a categorical variable, then it may create trouble while building the model. So it is necessary to encode these categorical variables into numbers.

### For Country variable:

Firstly, we will convert the country variables into categorical data. So to do this, we will use **LabelEncoder()** class from **preprocessing** library.

1. #Categorical data
2. #for Country Variable
3. from sklearn.preprocessing import LabelEncoder
4. label\_encoder\_x= LabelEncoder()
5. x[:, 0]= label\_encoder\_x.fit\_transform(x[:, 0])

### Output:

```
Out[15]:
array([[2, 38.0, 68000.0],
       [0, 43.0, 45000.0],
       [1, 30.0, 54000.0],
       [0, 48.0, 65000.0],
       [1, 40.0, 65222.22222222222],
       [2, 35.0, 58000.0],
       [1, 41.111111111111114, 53000.0],
       [0, 49.0, 79000.0],
       [2, 50.0, 88000.0],
       [0, 37.0, 77000.0]], dtype=object)
```

### Explanation:

In above code, we have imported **LabelEncoder** class of **sklearn library**. This class has successfully encoded the variables into digits.

But in our case, there are three country variables, and as we can see in the above output, these variables are encoded into 0, 1, and 2. By these values, the machine learning model may assume that there is some correlation between these variables which will produce the wrong output. So to remove this issue, we will use **dummy encoding**.

### Dummy Variables:

Dummy variables are those variables which have values 0 or 1. The 1 value gives the presence of that variable in a particular column, and rest variables become 0. With dummy encoding, we will have a number of columns equal to the number of categories.

In our dataset, we have 3 categories so it will produce three columns having 0 and 1 values. For Dummy Encoding, we will use **OneHotEncoder** class of **preprocessing** library.



1. #for Country Variable
2. from sklearn.preprocessing import LabelEncoder, OneHotEncoder
3. label\_encoder\_x= LabelEncoder()
4. x[:, 0]= label\_encoder\_x.fit\_transform(x[:, 0])
5. #Encoding for dummy variables
6. onehot\_encoder= OneHotEncoder(categorical\_features= [0])
7. x= onehot\_encoder.fit\_transform(x).toarray()

### Output:

```
array([[0.00000000e+00, 0.00000000e+00, 1.00000000e+00, 3.80000000e+01,
        6.80000000e+04],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 4.30000000e+01,
        4.50000000e+04],
       [0.00000000e+00, 1.00000000e+00, 0.00000000e+00, 3.00000000e+01,
        5.40000000e+04],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 4.80000000e+01,
        6.50000000e+04],
       [0.00000000e+00, 1.00000000e+00, 0.00000000e+00, 4.00000000e+01,
        6.52222222e+04],
       [0.00000000e+00, 0.00000000e+00, 1.00000000e+00, 3.50000000e+01,
        5.80000000e+04],
       [0.00000000e+00, 1.00000000e+00, 0.00000000e+00, 4.11111111e+01,
        5.30000000e+04],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 4.90000000e+01,
        7.90000000e+04],
       [0.00000000e+00, 0.00000000e+00, 1.00000000e+00, 5.00000000e+01,
        8.80000000e+04],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, 3.70000000e+01,
        7.70000000e+04]])
```

As we can see in the above output, all the variables are encoded into numbers 0 and 1 and divided into three columns.

It can be seen more clearly in the variables explorer section, by clicking on x option as:

	0	1	2	3	4
0	0	0	1	38	68000
1	1	0	0	43	45000
2	0	1	0	30	54000
3	1	0	0	48	65000
4	0	1	0	40	65222.2
5	0	0	1	35	58000
6	0	1	0	41.1111	53000
7	1	0	0	49	79000
8	0	0	1	50	88000
9	1	0	0	37	77000

#### For Purchased Variable:

1. `labelencoder_y= LabelEncoder()`
2. `y= labelencoder_y.fit_transform(y)`

For the second categorical variable, we will only use `labelencoder` object of **LabelEncoder** class. Here we are not using **OneHotEncoder** class because the purchased variable has only two categories yes or no, and which are automatically encoded into 0 and 1.

#### Output:

```
Out[17]: array([0, 1, 0, 0, 1, 1, 0, 1, 0, 1])
```

It can also be seen as:

y - NumPy array	
	0
0	0
1	1
2	0
3	0
4	1
5	1
6	0
7	1
8	0
9	1

Format    Resize    ☒ Background color

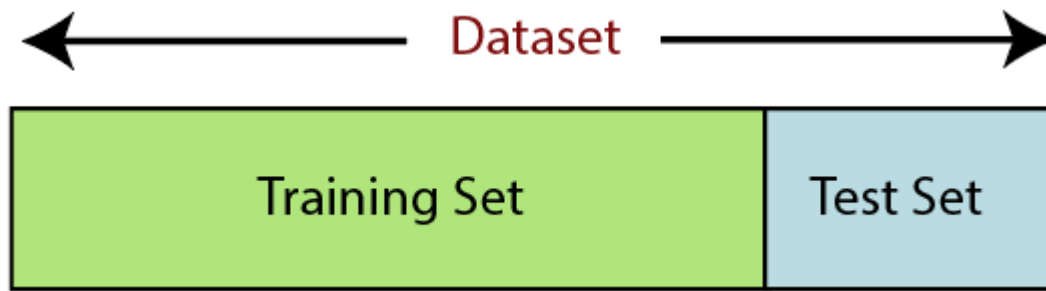
Save and Close    Close

## 6) Splitting the Dataset into the Training set and Test set

In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model.

Suppose, if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



**Training Set:** A subset of dataset to train the machine learning model, and we already know the output.

**Test set:** A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

For splitting the dataset, we will use the below lines of code:

1. `from sklearn.model_selection import train_test_split`
2. `x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.2, random_state=0)`

#### Explanation:

- In the above code, the first line is used for splitting arrays of the dataset into random train and test subsets.
- In the second line, we have used four variables for our output that are
  - **x\_train:** features for the training data
  - **x\_test:** features for testing data
  - **y\_train:** Dependent variables for training data
  - **y\_test:** Independent variable for testing data
- In **train\_test\_split() function**, we have passed four parameters in which first two are for arrays of data, and **test\_size** is for specifying the size of the test set. The test\_size maybe .5, .3, or .2, which tells the dividing ratio of training and testing sets.
- The last parameter **random\_state** is used to set a seed for a random generator so that you always get the same result, and the most used value for this is 42.

#### Output:

By executing the above code, we will get 4 different variables, which can be seen under the variable explorer section.

Variable explorer			
Name	Type	Size	Value
data_set	DataFrame	(10, 4)	Column names: Country, Age, Salary, Purchased
x	float64	(10, 5)	[[0.0e+00 0.0e+00 1.0e+00 3.8e+01 6.8e+04] [1.0e+00 0.0e+00 0.0e+00 4 ...
x_test	float64	(2, 5)	[[0.0e+00 1.0e+00 0.0e+00 3.0e+01 5.4e+04] [0.0e+00 0.0e+00 1.0e+00 5 ...
x_train	float64	(8, 5)	[[0.00000000e+00 1.00000000e+00 0.00000000e+00 4.00000000e+01 6.5222 ...
y	int32	(10,)	[0 1 0 0 1 1 0 1 0 1]
y_test	int32	(2,)	[0 0]
y_train	int32	(8,)	[1 1 1 0 1 0 0 1]

As we can see in the above image, the x and y variables are divided into 4 different variables with corresponding values.

## 7) Feature Scaling

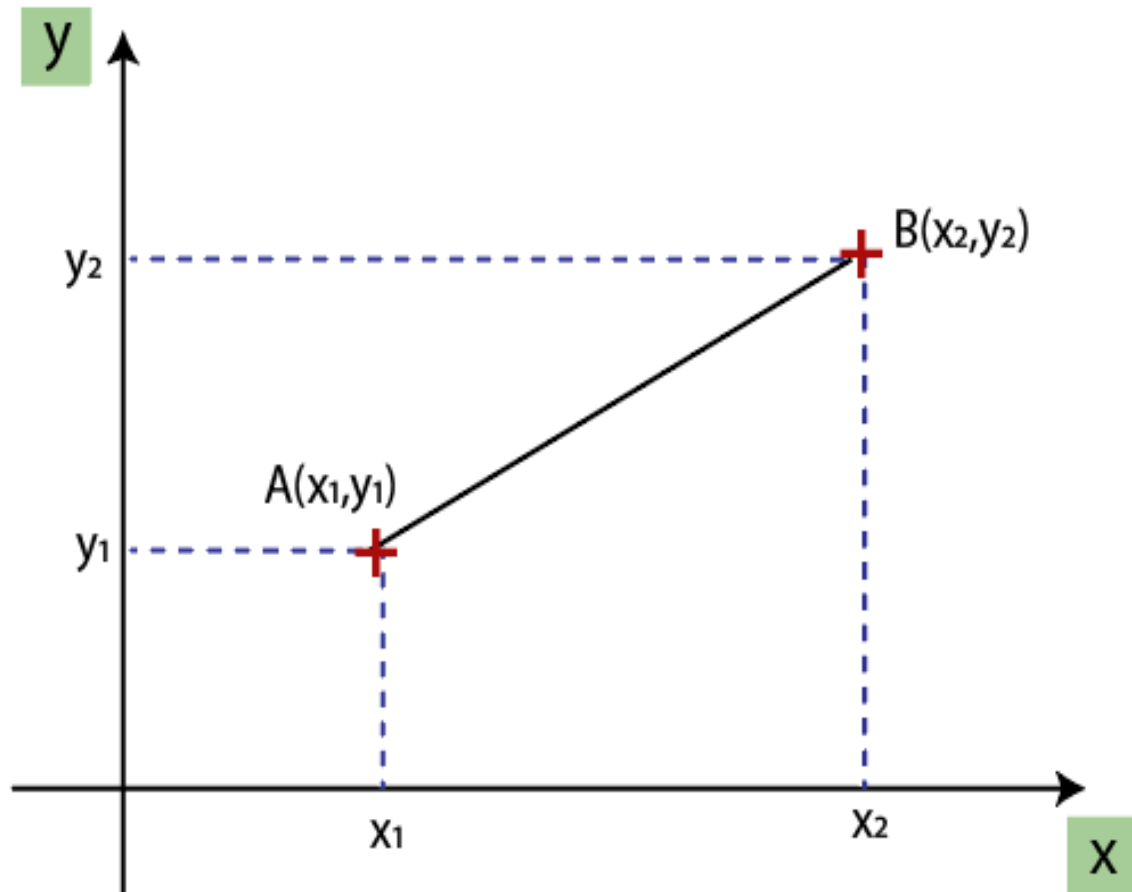
Feature scaling is the final step of data preprocessing in machine learning. It is a technique to standardize the independent variables of the dataset in a specific range. In feature scaling, we put our variables in the same range and in the same scale so that no any variable dominate the other variable.

Consider the below dataset:

data_set - DataFrame					
Index	Country	Age	Salary	Purchased	
0	India	38	68000	No	
1	France	43	45000	Yes	
2	Germany	30	54000	No	
3	France	48	65000	No	
4	Germany	40	nan	Yes	
5	India	35	58000	Yes	
6	Germany	nan	53000	No	
7	France	49	79000	Yes	
8	India	50	88000	No	
9	France	37	77000	Yes	

As we can see, the age and salary column values are not on the same scale. A machine learning model is based on **Euclidean distance**, and if we do not scale the variable, then it will cause some issue in our machine learning model.

Euclidean distance is given as:

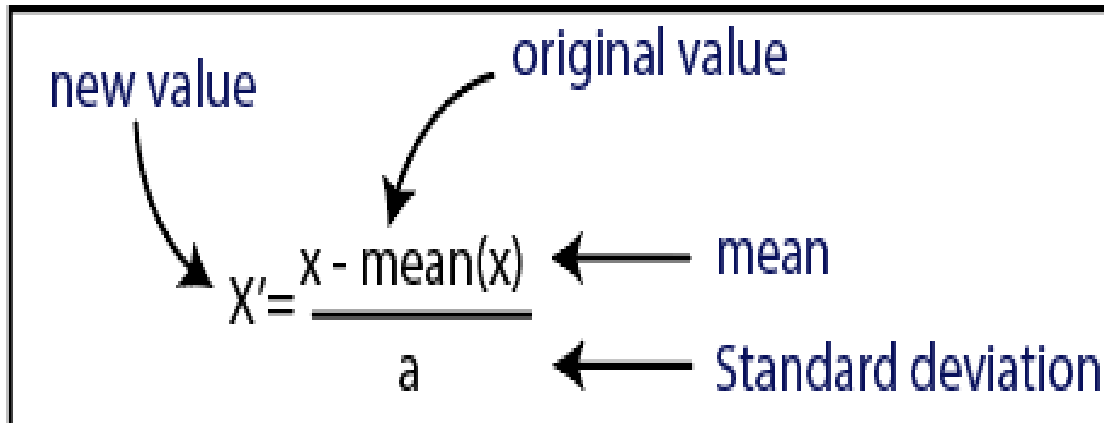


$$\text{Euclidean Distance Between A and B} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

If we compute any two values from age and salary, then salary values will dominate the age values, and it will produce an incorrect result. So to remove this issue, we need to perform feature scaling for machine learning.

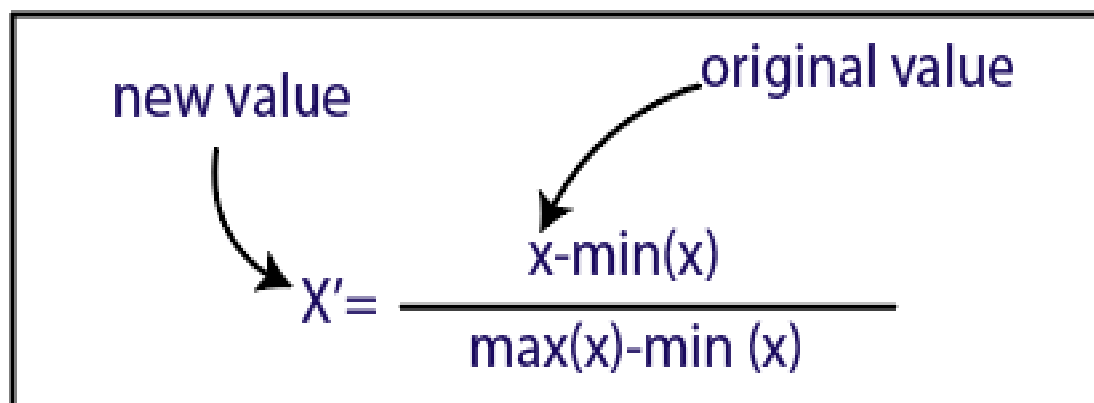
There are two ways to perform feature scaling in machine learning:

### Standardization



The diagram shows the formula for standardization: 
$$X' = \frac{x - \text{mean}(x)}{a}$$
 Arrows indicate the components: 'new value' points to  $X'$ , 'original value' points to  $x$ , 'mean' points to  $\text{mean}(x)$ , and 'Standard deviation' points to  $a$ .

### Normalization



The diagram shows the formula for normalization: 
$$X' = \frac{x - \min(x)}{\max(x) - \min(x)}$$
 Arrows indicate the components: 'new value' points to  $X'$ , 'original value' points to  $x$ , and the denominator is  $\max(x) - \min(x)$ .

Here, we will use the standardization method for our dataset.

For feature scaling, we will import **StandardScaler** class of **sklearn.Pre-processing** library as:

1. `from sklearn.preprocessing import StandardScaler`

Now, we will create the object of **StandardScaler** class for independent variables or features. And then we will fit and transform the training dataset.

1. `st_x= StandardScaler()`
2. `x_train= st_x.fit_transform(x_train)`

For test dataset, we will directly apply **transform()** function instead of **fit\_transform()** because it is already done in training set.

```
1. x_test= st_x.transform(x_test)
```

### Output:

By executing the above lines of code, we will get the scaled values for x\_train and x\_test as:

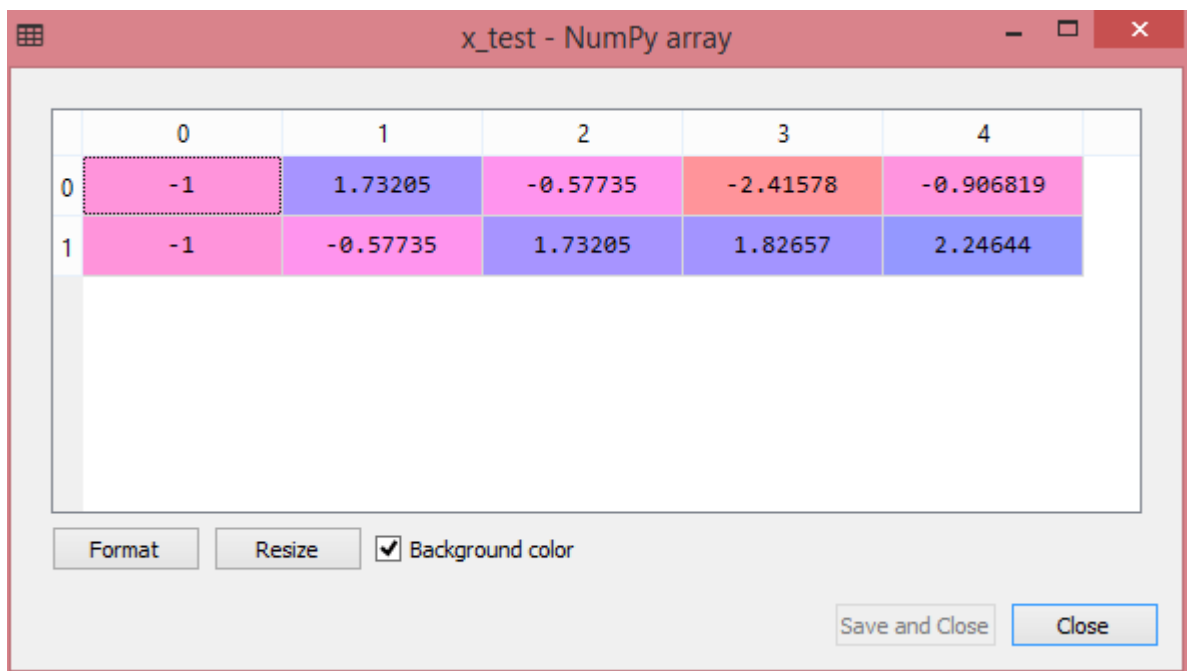
**x\_train:**



	0	1	2	3	4
0	-1	1.73205	-0.57735	-0.294607	0.133962
1	1	-0.57735	-0.57735	-0.930959	1.22627
2	1	-0.57735	-0.57735	0.341745	-1.7415
3	-1	1.73205	-0.57735	-0.0589215	-0.999562
4	1	-0.57735	-0.57735	1.61445	1.41175
5	1	-0.57735	-0.57735	1.40233	0.113352
6	-1	-0.57735	1.73205	-0.718842	0.391581
7	-1	-0.57735	1.73205	-1.35519	-0.535848



**x\_test:**



	0	1	2	3	4
0	-1	1.73205	-0.57735	-2.41578	-0.906819
1	-1	-0.57735	1.73205	1.82657	2.24644

As we can see in the above output, all the variables are scaled between values -1 to 1.

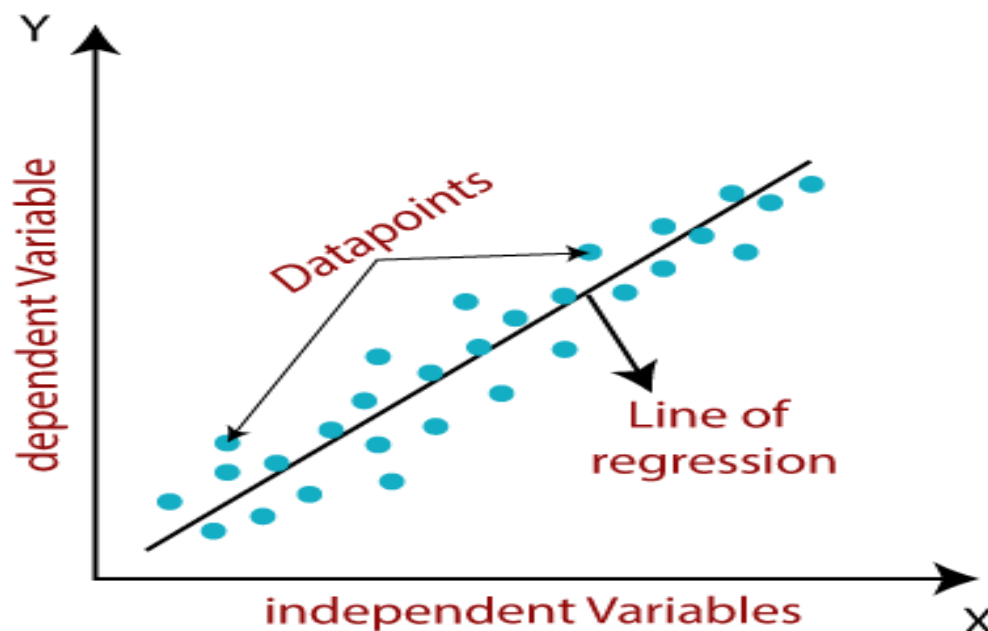
**Note:** Here, we have not scaled the dependent variable because there are only two values 0 and 1. But if these variables will have more range of values, then we will also need to scale those variables.

# LINEAR REGRESSION

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price**, etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (x) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:



Mathematically, we can represent a linear regression as:

$$y = a_0 + a_1x + \epsilon$$

Here,

Y= Dependent Variable (Target Variable)

X= Independent Variable (predictor Variable)

$a_0$ = intercept of the line (Gives an additional degree of freedom)

$a_1$  = Linear regression coefficient (scale factor to each input value).

$\varepsilon$  = random error

The values for x and y variables are training datasets for Linear Regression model representation.

## Types of Linear Regression

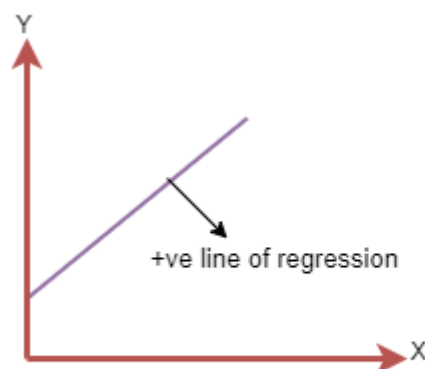
Linear regression can be further divided into two types of the algorithm:

- **Simple Linear Regression:**  
If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.
- **Multiple Linear regression:**  
If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

## Linear Regression Line

A linear line showing the relationship between the dependent and independent variables is called a **regression line**. A regression line can show two types of relationship:

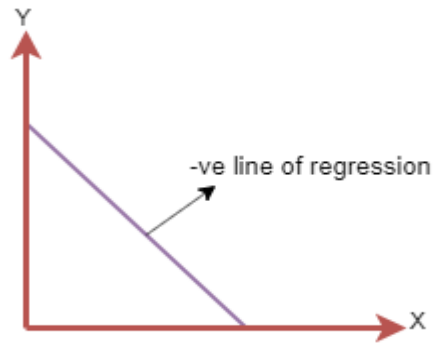
- **Positive Linear Relationship:**  
If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.



The line equation will be:  $Y = a_0 + a_1X$

- **Negative Linear Relationship:**  
If the dependent variable decreases on the Y-axis and independent variable

increases on the X-axis, then such a relationship is called a negative linear relationship.



The line of equation will be:  $Y = -a_0 + a_1x$

### Finding the best fit line:

When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error.

The different values for weights or the coefficient of lines ( $a_0$ ,  $a_1$ ) gives a different line of regression, so we need to calculate the best values for  $a_0$  and  $a_1$  to find the best fit line, so to calculate this we use cost function.

### Cost function-

- The different values for weights or coefficient of lines ( $a_0$ ,  $a_1$ ) gives the different line of regression, and the cost function is used to estimate the values of the coefficient for the best fit line.
- Cost function optimizes the regression coefficients or weights. It measures how a linear regression model is performing.
- We can use the cost function to find the accuracy of the **mapping function**, which maps the input variable to the output variable. This mapping function is also known as **Hypothesis function**.

For Linear Regression, we use the **Mean Squared Error (MSE)** cost function, which is the average of squared error occurred between the predicted values and actual values. It can be written as:

For the above linear equation, MSE can be calculated as:

$$MSE = \frac{1}{N} \sum_{i=1}^n (y_i - (a_1x_i + a_0))^2$$

**Where,**

N=Total number of observation

$Y_i$  = Actual value

$(a_1x_i+a_0)$ = Predicted value.

**Residuals:** The distance between the actual value and predicted values is called residual. If the observed points are far from the regression line, then the residual will be high, and so cost function will high. If the scatter points are close to the regression line, then the residual will be small and hence the cost function.

### **Gradient Descent:**

- Gradient descent is used to minimize the MSE by calculating the gradient of the cost function.
- A regression model uses gradient descent to update the coefficients of the line by reducing the cost function.
- It is done by a random selection of values of coefficient and then iteratively update the values to reach the minimum cost function.

### **Model Performance:**

The Goodness of fit determines how the line of regression fits the set of observations. The process of finding the best model out of various models is called **optimization**. It can be achieved by below method:

#### **1. R-squared method:**

- R-squared is a statistical method that determines the goodness of fit.
- It measures the strength of the relationship between the dependent and independent variables on a scale of 0-100%.
- The high value of R-square determines the less difference between the predicted values and actual values and hence represents a good model.
- It is also called a **coefficient of determination**, or **coefficient of multiple determination** for multiple regression.
- It can be calculated from the below formula:

$$\text{R-squared} = \frac{\text{Explained variation}}{\text{Total Variation}}$$

## Assumptions of Linear Regression

Below are some important assumptions of Linear Regression. These are some formal checks while building a Linear Regression model, which ensures to get the best possible result from the given dataset.

- **Linear relationship between the features and target:**  
Linear regression assumes the linear relationship between the dependent and independent variables.
- **Small or no multicollinearity between the features:**  
Multicollinearity means high-correlation between the independent variables. Due to multicollinearity, it may difficult to find the true relationship between the predictors and target variables. Or we can say, it is difficult to determine which predictor variable is affecting the target variable and which is not. So, the model assumes either little or no multicollinearity between the features or independent variables.
- **Homoscedasticity Assumption:**  
Homoscedasticity is a situation when the error term is the same for all the values of independent variables. With homoscedasticity, there should be no clear pattern distribution of data in the scatter plot.
- **Normal distribution of error terms:**  
Linear regression assumes that the error term should follow the normal distribution pattern. If error terms are not normally distributed, then confidence intervals will become either too wide or too narrow, which may cause difficulties in finding coefficients.  
It can be checked using the **q-q plot**. If the plot shows a straight line without any deviation, which means the error is normally distributed.
- **No autocorrelations:**  
The linear regression model assumes no autocorrelation in error terms. If there will be any correlation in the error term, then it will drastically reduce the accuracy of the model. Autocorrelation usually occurs if there is a dependency between residual errors.

## Simple Linear Regression in Machine Learning

Simple Linear Regression is a type of Regression algorithms that models the relationship between a dependent variable and a single independent variable. The relationship shown by a Simple Linear Regression model is linear or a sloped straight line, hence it is called Simple Linear Regression.

The key point in Simple Linear Regression is that the ***dependent variable must be a continuous/real value***. However, the independent variable can be measured on continuous or categorical values.

Simple Linear regression algorithm has mainly two objectives:

- **Model the relationship between the two variables.** Such as the relationship between Income and expenditure, experience and Salary, etc.
- **Forecasting new observations.** Such as Weather forecasting according to temperature, Revenue of a company according to the investments in a year, etc.

## Simple Linear Regression Model:

The Simple Linear Regression model can be represented using the below equation:

$$y = a_0 + a_1X + \epsilon$$

Where,

$a_0$  = It is the intercept of the Regression line (can be obtained putting  $x=0$ )

$a_1$  = It is the slope of the regression line, which tells whether the line is increasing or decreasing.

$\epsilon$  = The error term. (For a good model it will be negligible)

## Multiple Linear Regression

In the previous topic, we have learned about Simple Linear Regression, where a single Independent/Predictor(X) variable is used to model the response variable (Y). But there may be various cases in which the response variable is affected by more than one predictor variable; for such cases, the Multiple Linear Regression algorithm is used.

Moreover, Multiple Linear Regression is an extension of Simple Linear regression as it takes more than one predictor variable to predict the response variable. We can define it as:

***Multiple Linear Regression is one of the important regression algorithms which models the linear relationship between a single dependent continuous variable and more than one independent variable.***

### Example:

Prediction of CO<sub>2</sub> emission based on engine size and number of cylinders in a car.

## Some key points about MLR:

- For MLR, the dependent or target variable(Y) must be the continuous/real, but the predictor or independent variable may be of continuous or categorical form.
- Each feature variable must model the linear relationship with the dependent variable.
- MLR tries to fit a regression line through a multidimensional space of data-points.

## MLR equation:

In Multiple Linear Regression, the target variable(Y) is a linear combination of multiple predictor variables  $x_1, x_2, x_3, \dots, x_n$ . Since it is an enhancement of Simple Linear Regression, so the same is applied for the multiple linear regression equation, the equation becomes:

$$Y = b_0 \cdot X_0 + b_1 \cdot X_1 + b_2 \cdot X_2 + \dots + b_n \cdot X_n \dots \dots \dots (2)$$

Where,

Y= Output/Response variable

$b_0, b_1, b_2, b_3, b_n, \dots$  = Coefficients of the model.

$x_1, x_2, x_3, x_4, \dots$  = Various Independent/feature variable

## Assumptions for Multiple Linear Regression:

- A **linear relationship** should exist between the Target and predictor variables.
- The regression residuals must be **normally distributed**.
- MLR assumes little or **no multicollinearity** (correlation between the independent variable) in data.

## Applications of Multiple Linear Regression:

There are mainly two applications of Multiple Linear Regression:

- Effectiveness of Independent variable on prediction:
- Predicting the impact of changes:

## What is Backward Elimination?

Backward elimination is a feature selection technique while building a machine learning model. It is used to remove those features that do not have a significant effect



on the dependent variable or prediction of output. There are various ways to build a model in Machine Learning, which are:

1. All-in
2. Backward Elimination
3. Forward Selection
4. Bidirectional Elimination
5. Score Comparison

Above are the possible methods for building the model in Machine learning, but we will only use here the Backward Elimination process as it is the fastest method.

### **Steps of Backward Elimination**

Below are some main steps which are used to apply backward elimination process:

**Step-1:** Firstly, We need to select a significance level to stay in the model. (SL=0.05)

**Step-2:** Fit the complete model with all possible predictors/independent variables.

**Step-3:** Choose the predictor which has the highest P-value, such that.

- a. If P-value > SL, go to step 4.
- b. Else Finish, and Our model is ready.

**Step-4:** Remove that predictor.

**Step-5:** Rebuild and fit the model with the remaining variables.

### **Need for Backward Elimination: An optimal Multiple Linear Regression model:**

In the previous chapter, we discussed and successfully created our Multiple Linear Regression model, where we took **4 independent variables (R&D spend, Administration spend, Marketing spend, and state (dummy variables))** and **one dependent variable (Profit)**. But that model is not optimal, as we have included all the independent variables and do not know which independent model is most affecting and which one is the least affecting for the prediction.

Unnecessary features increase the complexity of the model. Hence it is good to have only the most significant features and keep our model simple to get the better result.

So, in order to optimize the performance of the model, we will use the Backward Elimination method. This process is used to optimize the performance of the MLR

model as it will only include the most affecting feature and remove the least affecting feature. Let's start to apply it to our MLR model.

## ML Polynomial Regression

- Polynomial Regression is a regression algorithm that models the relationship between a dependent(y) and independent variable(x) as nth degree polynomial. The Polynomial Regression equation is given below:

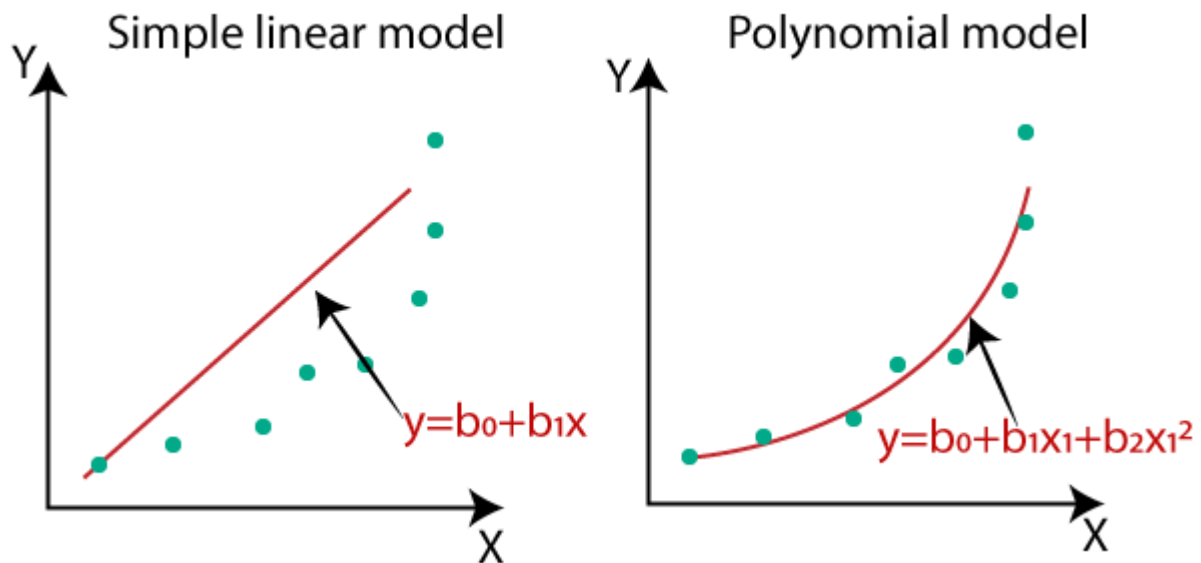
$$y = b_0 + b_1x_1 + b_2x_1^2 + b_3x_1^3 + \dots + b_nx_1^n$$

- It is also called the special case of Multiple Linear Regression in ML. Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression.
- It is a linear model with some modification in order to increase the accuracy.
- The dataset used in Polynomial regression for training is of non-linear nature.
- It makes use of a linear regression model to fit the complicated and non-linear functions and datasets.
- **Hence, "In Polynomial regression, the original features are converted into Polynomial features of required degree (2,3,..,n) and then modeled using a linear model."**

## Need for Polynomial Regression:

The need of Polynomial Regression in ML can be understood in the below points:

- If we apply a linear model on a **linear dataset**, then it provides us a good result as we have seen in Simple Linear Regression, but if we apply the same model without any modification on a **non-linear dataset**, then it will produce a drastic output. Due to which loss function will increase, the error rate will be high, and accuracy will be decreased.
- So for such cases, **where data points are arranged in a non-linear fashion, we need the Polynomial Regression model**. We can understand it in a better way using the below comparison diagram of the linear dataset and non-linear dataset.
- In the below image, we have taken a dataset which is arranged non-linearly. So if we try to cover it with a linear model, then we can clearly see that it hardly covers any data point. On the other hand, a curve is suitable to cover most of the data points, which is of the Polynomial model.
- Hence, *if the datasets are arranged in a non-linear fashion, then we should use the Polynomial Regression model instead of Simple Linear Regression.*



**Note:** A Polynomial Regression algorithm is also called Polynomial Linear Regression because it does not depend on the variables, instead, it depends on the coefficients, which are arranged in a linear fashion.

### Equation of the Polynomial Regression Model:

**Simple Linear Regression equation:**

$$y = b_0 + b_1x \quad \text{.....(a)}$$

**Multiple Linear Regression equation:**

$$y = b_0 + b_1x + b_2x_2 + b_3x_3 + \dots + b_nx_n \quad \text{.....(b)}$$

**Polynomial Regression equation:**

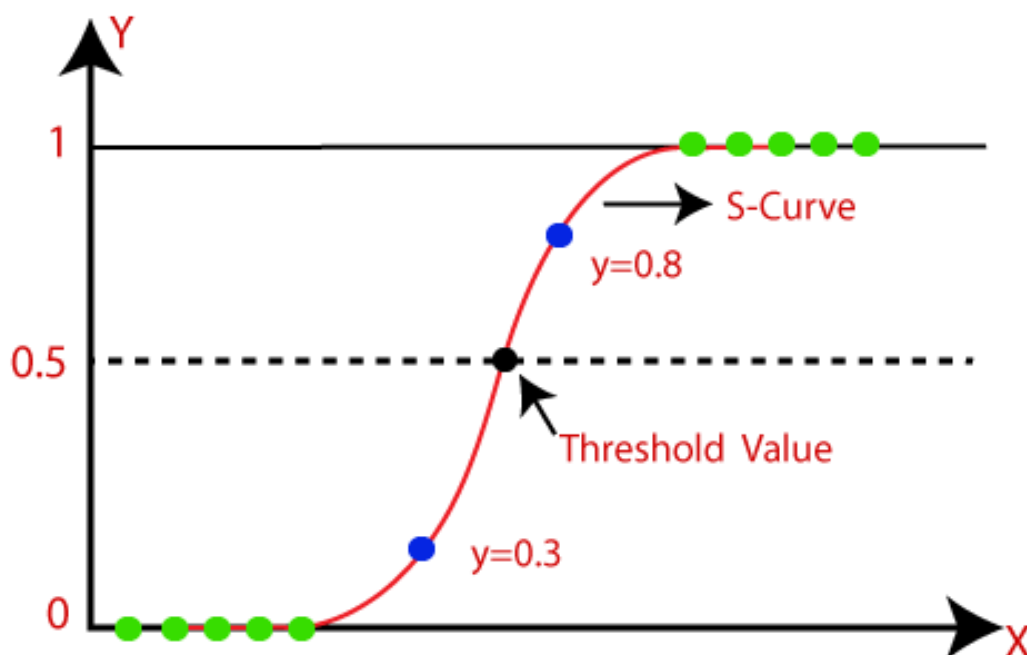
$$y = b_0 + b_1x + b_2x^2 + b_3x^3 + \dots + b_nx^n \quad \text{.....(c)}$$

When we compare the above three equations, we can clearly see that all three equations are Polynomial equations but differ by the degree of variables. The Simple and Multiple Linear equations are also Polynomial equations with a single degree, and the Polynomial regression equation is Linear equation with the nth degree. So if we add a degree to our linear equations, then it will be converted into Polynomial Linear equations.

**Note:** To better understand Polynomial Regression, you must have knowledge of Simple Linear Regression.

# LOGISTIC REGRESSION

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1.**
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems.**
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



**Note:** Logistic regression uses the concept of predictive modeling as regression; therefore, it is called logistic regression, but is used to classify samples; Therefore, it falls under the classification algorithm.

## Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

## Assumptions for Logistic Regression:

- The dependent variable must be categorical in nature.
- The independent variable should not have multi-collinearity.

## Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

- We know the equation of the straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

- $$\frac{y}{1-y}; 0 \text{ for } y=0, \text{ and infinity for } y=1$$

- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$\log \left[ \frac{y}{1-y} \right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

### **Type of Logistic Regression:**

On the basis of the categories, Logistic Regression can be classified into three types:

- **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

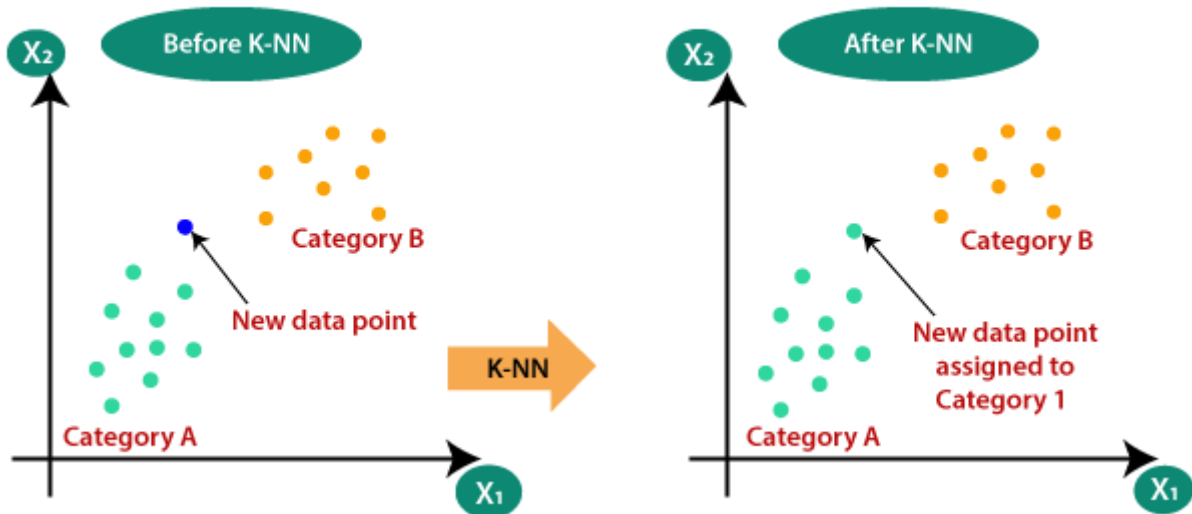
# K-NEAREST NEIGHBOUR(KNN) ALGORITHM

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
- **Example:** Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.



**Why do we need a K-NN Algorithm?**

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point  $x_1$ , so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



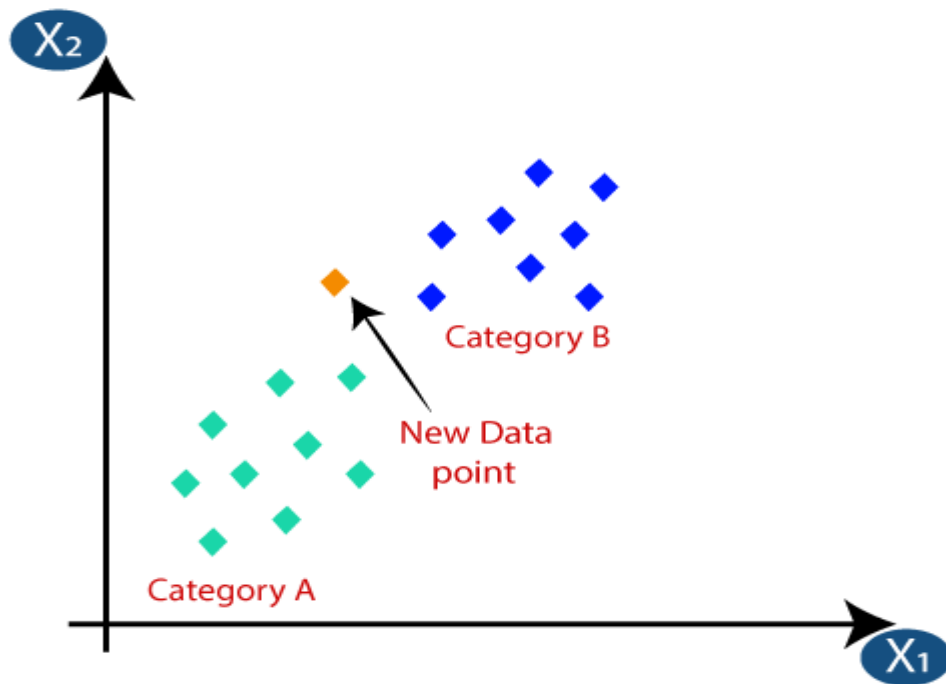
### How does K-NN work?

The K-NN working can be explained on the basis of the below algorithm:

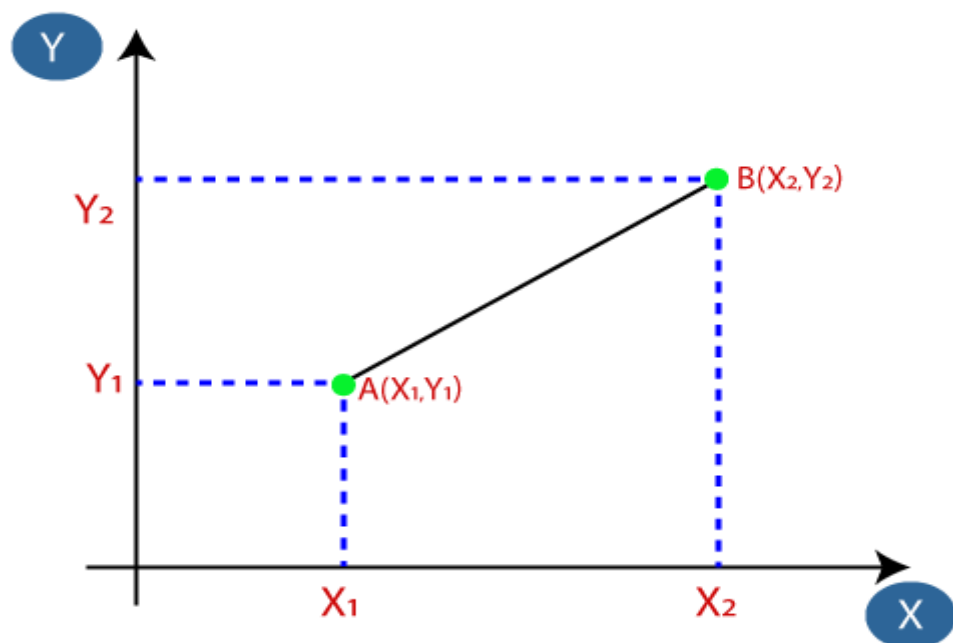
- **Step-1:** Select the number  $K$  of the neighbors
- **Step-2:** Calculate the Euclidean distance of  **$K$  number of neighbors**
- **Step-3:** Take the  $K$  nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these  $k$  neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:



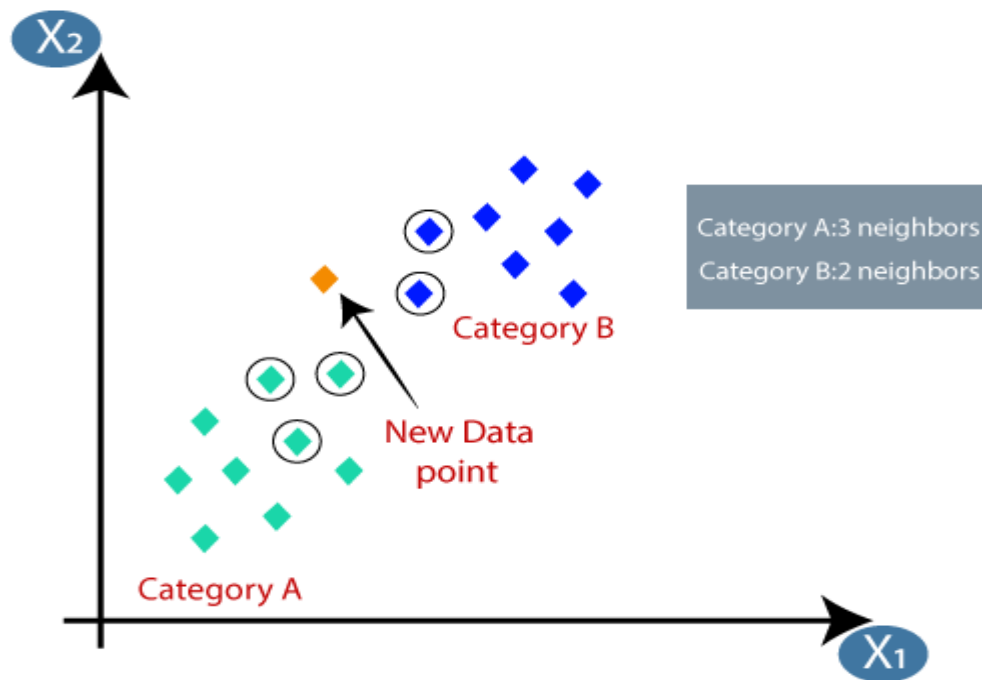


- Firstly, we will choose the number of neighbors, so we will choose the  $k=5$ .
- Next, we will calculate the **Euclidean distance** between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:



$$\text{Euclidean Distance between } A_1 \text{ and } B_2 = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$$

- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:



- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

### How to select the value of K in the K-NN Algorithm?

Below are some points to remember while selecting the value of K in the K-NN algorithm:

- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
- A very low value for K such as  $K=1$  or  $K=2$ , can be noisy and lead to the effects of outliers in the model.
- Large values for K are good, but it may find some difficulties.

### Advantages of KNN Algorithm:

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

### Disadvantages of KNN Algorithm:

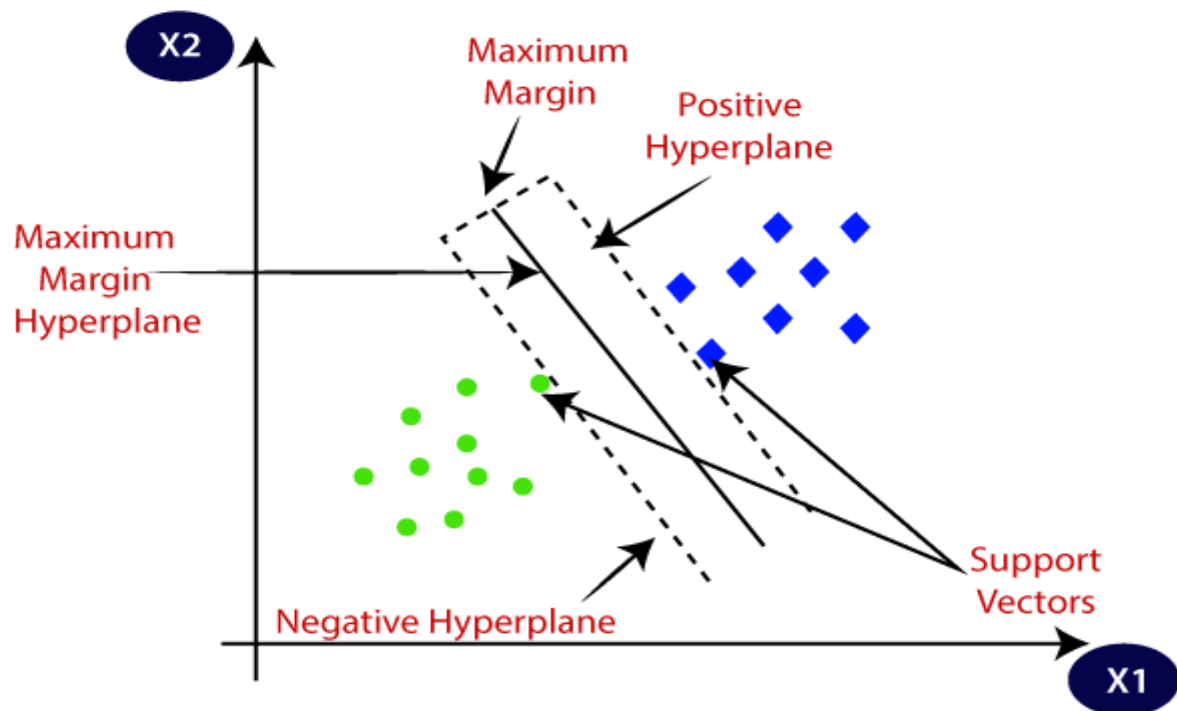
- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

# SUPPORT VECTOR MACHINE ALGORITHM

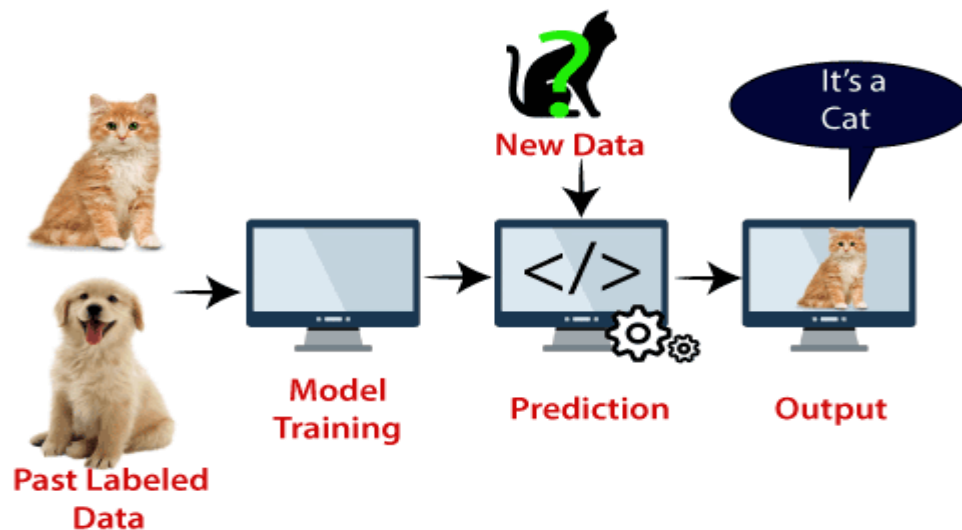
Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



**Example:** SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:



SVM algorithm can be used for **Face detection**, **image classification**, **text categorization**, etc.

## Types of SVM

**SVM can be of two types:**

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

## Hyperplane and Support Vectors in the SVM algorithm:

**Hyperplane:** There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

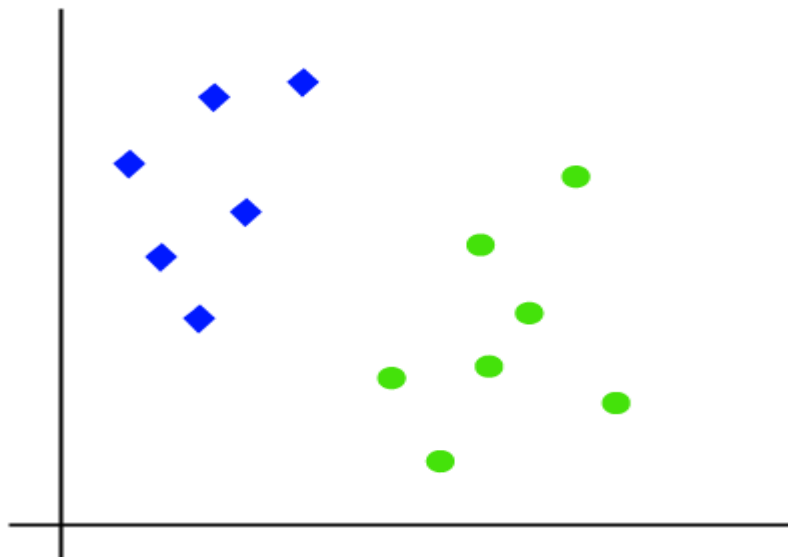
## Support Vectors:

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

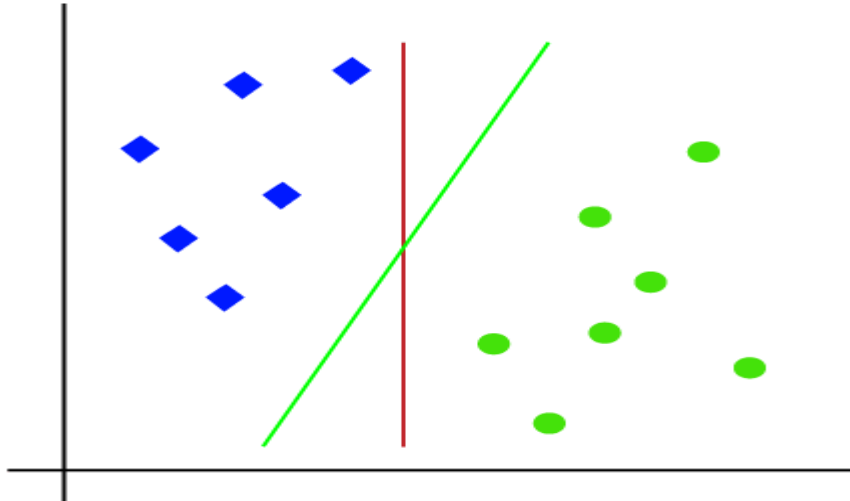
## How does SVM works?

### Linear SVM:

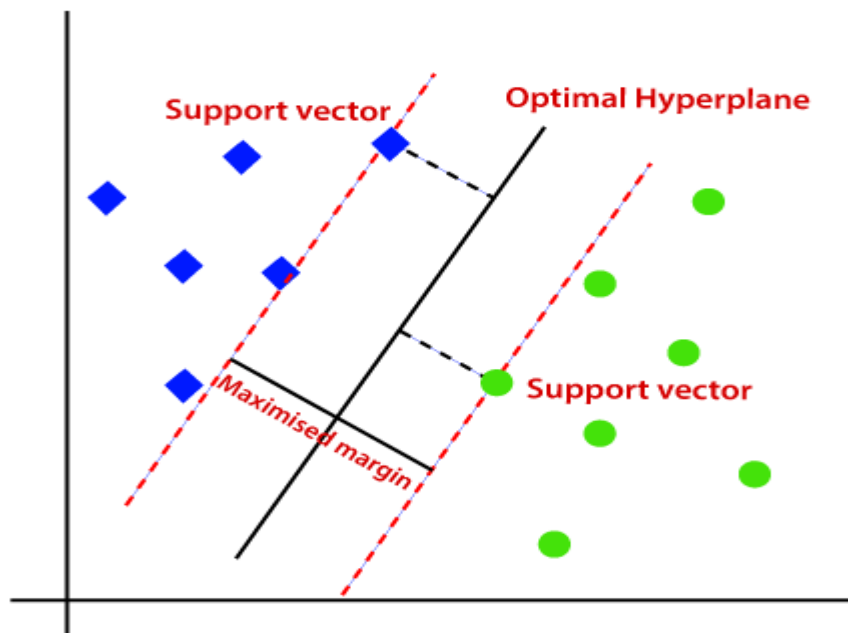
The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features  $x_1$  and  $x_2$ . We want a classifier that can classify the pair( $x_1$ ,  $x_2$ ) of coordinates in either green or blue. Consider the below image:



So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:

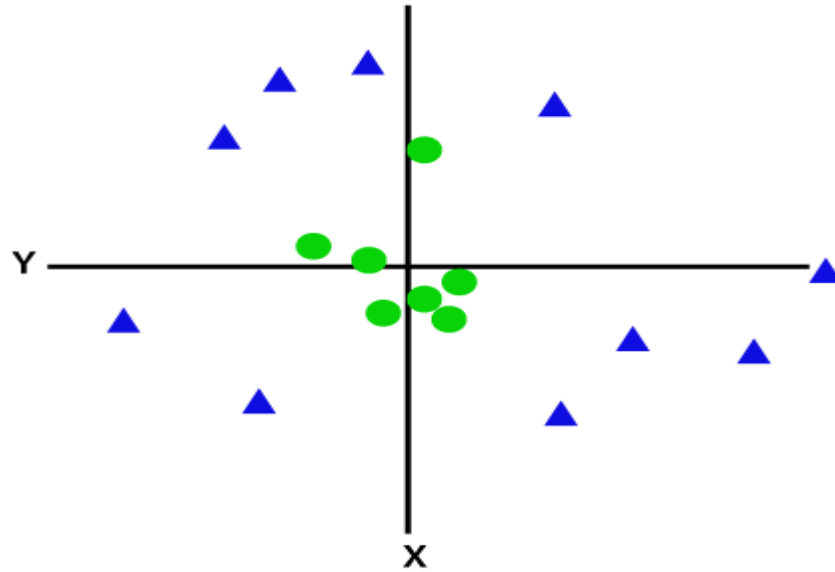


Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a **hyperplane**. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as **margin**. And the goal of SVM is to maximize this margin. The **hyperplane** with maximum margin is called the **optimal hyperplane**.



### Non-Linear SVM:

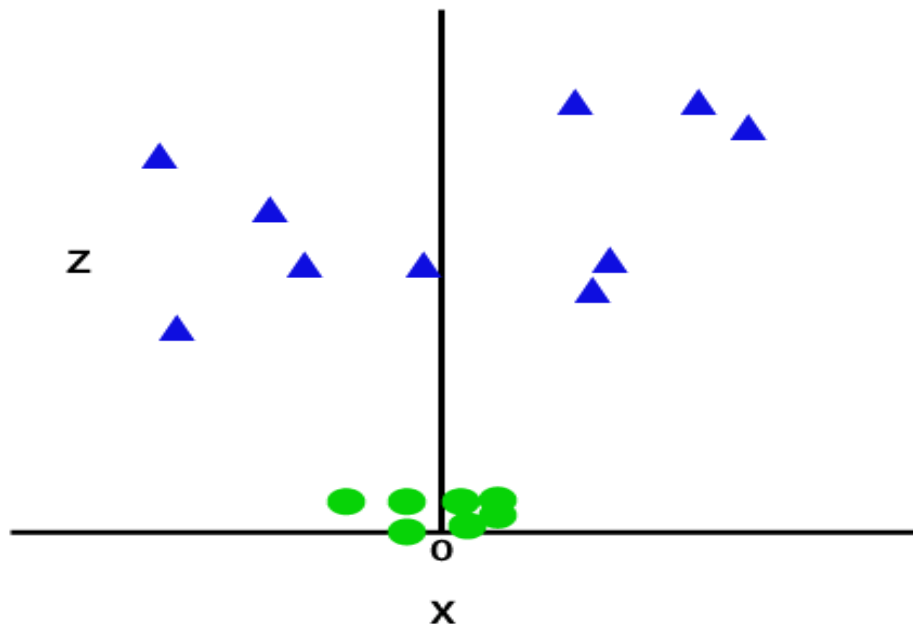
If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



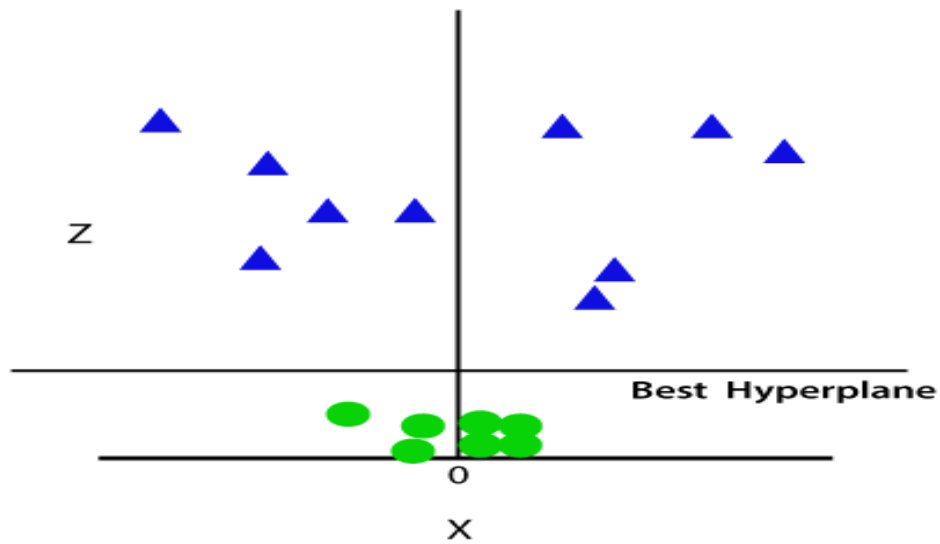
So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third dimension z. It can be calculated as:

$$z = x^2 + y^2$$

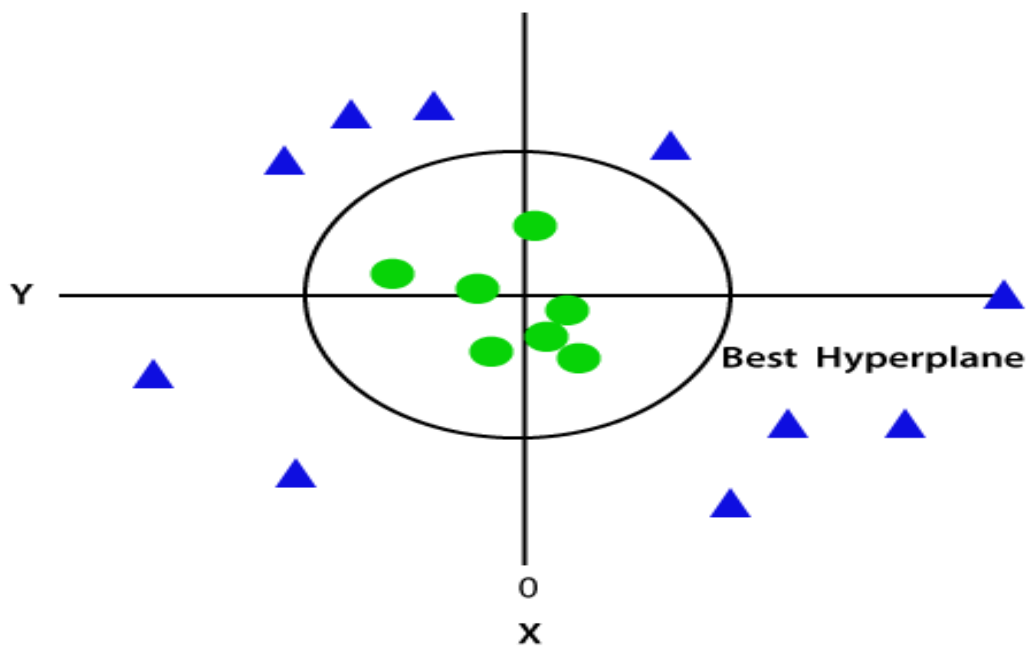
By adding the third dimension, the sample space will become as below image:



So now, SVM will divide the datasets into classes in the following way. Consider the below image:



Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with  $z=1$ , then it will become as:



Hence we get a circumference of radius 1 in case of non-linear data.



# NAÏVE BAYES CLASSIFIER ALGORITHM

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.
- It is mainly used in *text classification* that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- **It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.**
- Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentimental analysis, and classifying articles.**

## Why is it called Naïve Bayes?

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

- **Naïve:** It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- **Bayes:** It is called Bayes because it depends on the principle of Bayes' Theorem.

## Bayes' Theorem:

- Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Where,

**P(A|B) is Posterior probability:** Probability of hypothesis A on the observed event B.

**P(B|A) is Likelihood probability:** Probability of the evidence given that the probability of a hypothesis is true.

**P(A) is Prior Probability:** Probability of hypothesis before observing the evidence.

**P(B) is Marginal Probability:** Probability of Evidence.

### Working of Naïve Bayes' Classifier:

Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of **weather conditions** and corresponding target variable "**Play**". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.
3. Now, use Bayes theorem to calculate the posterior probability.

**Problem:** If the weather is sunny, then the Player should play or not?

**Solution:** To solve this, first consider the below dataset:

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes

11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

**Frequency table for the Weather Conditions:**

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	5

**Likelihood table weather condition:**

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

**Applying Bayes'theorem:**

$$P(\text{Yes}|\text{Sunny})= P(\text{Sunny}|\text{Yes}) * P(\text{Yes}) / P(\text{Sunny})$$

$$P(\text{Sunny}|\text{Yes})= 3/10= 0.3$$

$$P(\text{Sunny})= 0.35$$

$$P(\text{Yes})=0.71$$

$$\text{So } P(\text{Yes}|\text{Sunny}) = 0.3 * 0.71 / 0.35 = \mathbf{0.60}$$

$$P(\text{No}|\text{Sunny})= P(\text{Sunny}|\text{No}) * P(\text{No}) / P(\text{Sunny})$$

$$P(\text{Sunny}|\text{NO}) = 2/4 = 0.5$$

$$P(\text{No}) = 0.29$$

$$P(\text{Sunny}) = 0.35$$

$$\text{So } P(\text{No}|\text{Sunny}) = 0.5 * 0.29 / 0.35 = \mathbf{0.41}$$

So as we can see from the above calculation that  $P(\text{Yes}|\text{Sunny}) > P(\text{No}|\text{Sunny})$

**Hence on a Sunny day, Player can play the game.**

### **Advantages of Naïve Bayes Classifier:**

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for **text classification problems**.

### **Disadvantages of Naïve Bayes Classifier:**

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

### **Applications of Naïve Bayes Classifier:**

- It is used for **Credit Scoring**.
- It is used in **medical data classification**.
- It can be used in **real-time predictions** because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as **Spam filtering** and **Sentiment analysis**.

### **Types of Naïve Bayes Model:**

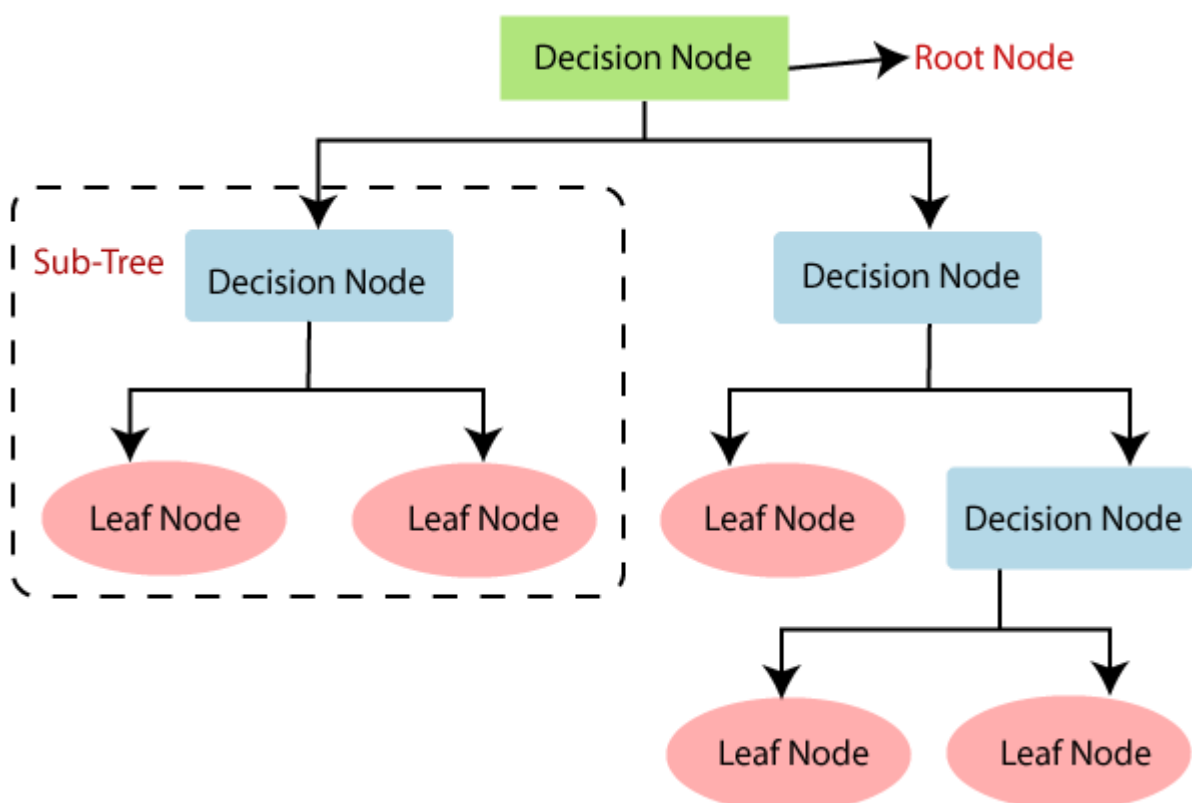
There are three types of Naive Bayes Model, which are given below:

- **Gaussian:** The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.

- **Multinomial:** The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc.  
The classifier uses the frequency of words for the predictors.
- **Bernoulli:** The Bernoulli classifier works similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables. Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

# DECISION TREE CLASSIFICATION ALGORITHM

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome**.
- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node**. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the **CART algorithm**, which stands for **Classification and Regression Tree algorithm**.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:



Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.

## Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

## Decision Tree Terminologies

- ✓ **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- ✓ **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- ✓ **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- ✓ **Branch/Sub Tree:** A tree formed by splitting the tree.
- ✓ **Pruning:** Pruning is the process of removing the unwanted branches from the tree.
- ✓ **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

## How does the Decision Tree algorithm Work?

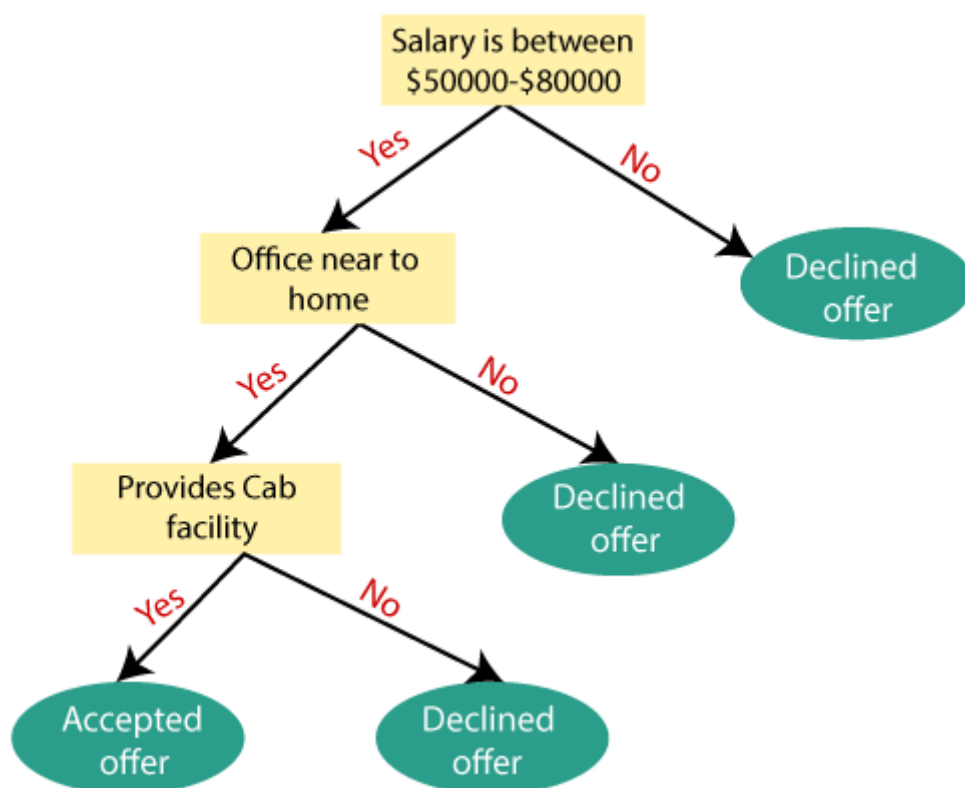
In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM)**.
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.

- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

**Example:** Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



## Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM**. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- **Information Gain**
- **Gini Index**



## 1. Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

$$1. \text{ Information Gain} = \text{Entropy}(S) - [(\text{Weighted Avg}) * \text{Entropy}(\text{each feature})]$$

**Entropy:** Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

$$\text{Entropy}(s) = -P(\text{yes}) \log_2 P(\text{yes}) - P(\text{no}) \log_2 P(\text{no})$$

**Where,**

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

## 2. Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

$$\text{Gini Index} = 1 - \sum_i P_i^2$$

## Pruning: Getting an Optimal Decision tree

*Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.*

A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a technique that decreases the size

of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree **pruning** technology used:

- **Cost Complexity Pruning**
- **Reduced Error Pruning.**

### **Advantages of the Decision Tree**

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

### **Disadvantages of the Decision Tree**

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the **Random Forest algorithm**.
- For more class labels, the computational complexity of the decision tree may increase.

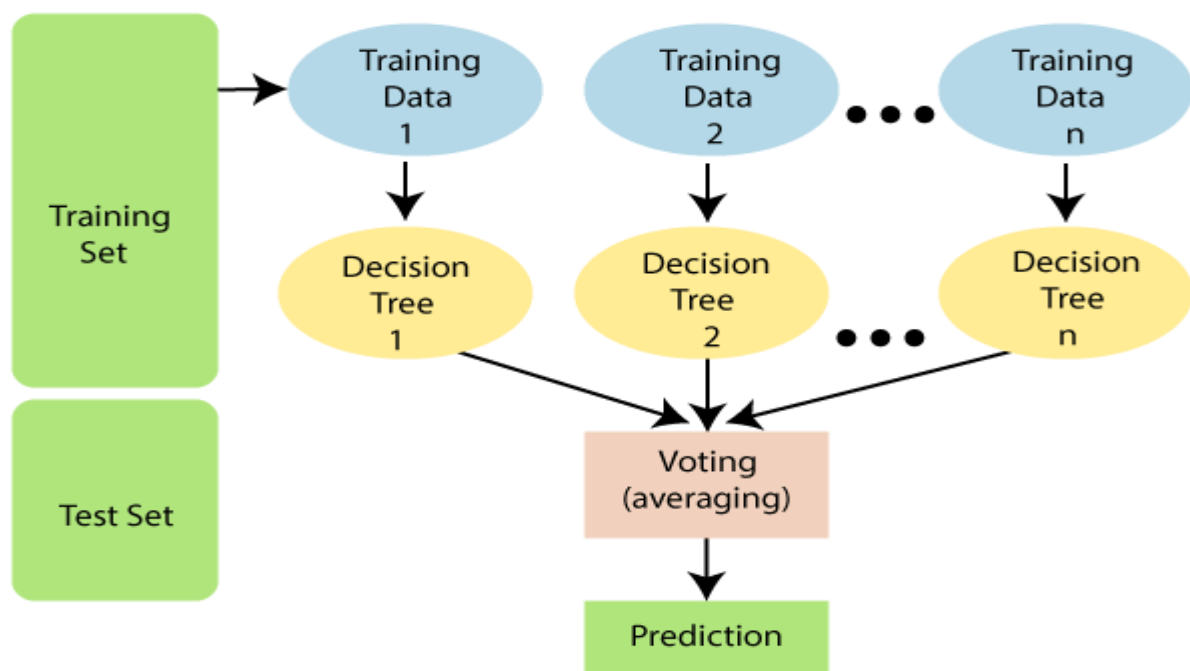
# RANDOM FOREST ALGORITHM

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning**, which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model*.

As the name suggests, **"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

**The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.**

The below diagram explains the working of the Random Forest algorithm:



Note: To better understand the Random Forest Algorithm, you should have knowledge of the Decision Tree Algorithm.

## Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may

not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

## Why use Random Forest?

Below are some points that explain why we should use the Random Forest algorithm:

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

## How does Random Forest algorithm work?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

**Step-1:** Select random K data points from the training set.

**Step-2:** Build the decision trees associated with the selected data points (Subsets).

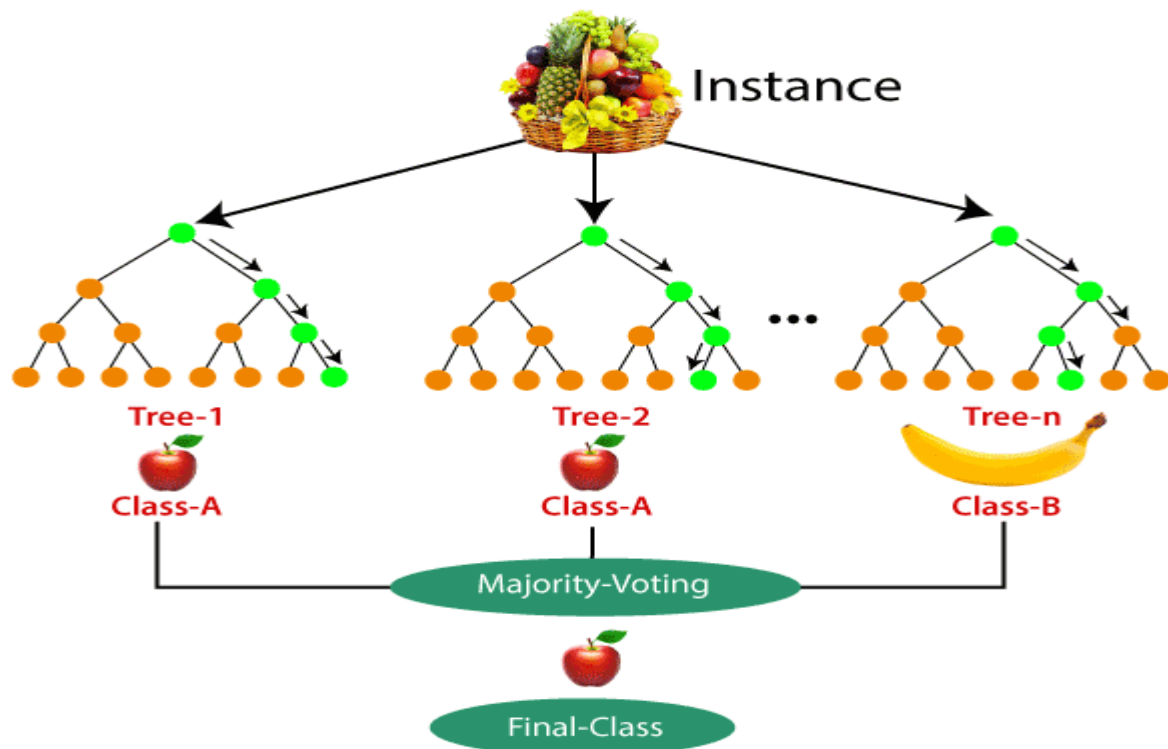
**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

The working of the algorithm can be better understood by the below example:

**Example:** Suppose there is a dataset that contains multiple fruit images. So, this dataset is given to the Random forest classifier. The dataset is divided into subsets and given to each decision tree. During the training phase, each decision tree produces a prediction result, and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision. Consider the below image:



## Applications of Random Forest

There are mainly four sectors where Random forest mostly used:

1. **Banking:** Banking sector mostly uses this algorithm for the identification of loan risk.
2. **Medicine:** With the help of this algorithm, disease trends and risks of the disease can be identified.
3. **Land Use:** We can identify the areas of similar land use by this algorithm.
4. **Marketing:** Marketing trends can be identified using this algorithm.

## Advantages of Random Forest

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

## Disadvantages of Random Forest

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

# CLUSTERING

Clustering or cluster analysis is a machine learning technique, which groups the unlabelled dataset. It can be defined as ***"A way of grouping the data points into different clusters, consisting of similar data points. The objects with the possible similarities remain in a group that has less or no similarities with another group."***

It does it by finding some similar patterns in the unlabelled dataset such as shape, size, color, behavior, etc., and divides them as per the presence and absence of those similar patterns.

It is an unsupervised learning method, hence no supervision is provided to the algorithm, and it deals with the unlabeled dataset.

After applying this clustering technique, each cluster or group is provided with a cluster-ID. ML system can use this id to simplify the processing of large and complex datasets.

The clustering technique is commonly used for **statistical data analysis**.

Note: Clustering is somewhere similar to the classification algorithm, but the difference is the type of dataset that we are using. In classification, we work with the labeled data set, whereas in clustering, we work with the unlabelled dataset.

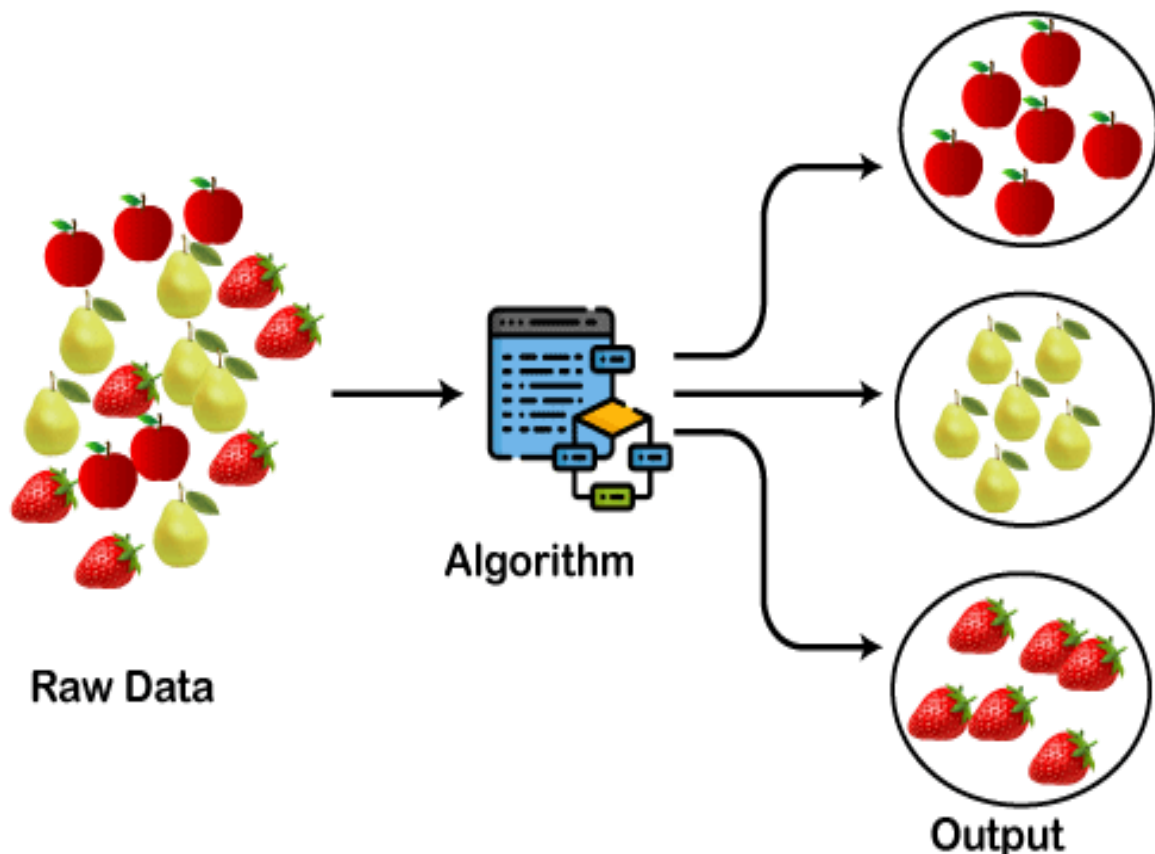
**Example:** Let's understand the clustering technique with the real-world example of Mall: When we visit any shopping mall, we can observe that the things with similar usage are grouped together. Such as the t-shirts are grouped in one section, and trousers are at other sections, similarly, at vegetable sections, apples, bananas, Mangoes, etc., are grouped in separate sections, so that we can easily find out the things. The clustering technique also works in the same way. Other examples of clustering are grouping documents according to the topic.

The clustering technique can be widely used in various tasks. Some most common uses of this technique are:

- Market Segmentation
- Statistical data analysis
- Social network analysis
- Image segmentation
- Anomaly detection, etc.

Apart from these general usages, it is used by the **Amazon** in its recommendation system to provide the recommendations as per the past search of products. **Netflix** also uses this technique to recommend the movies and web-series to its users as per the watch history.

The below diagram explains the working of the clustering algorithm. We can see the different fruits are divided into several groups with similar properties.



## Types of Clustering Methods

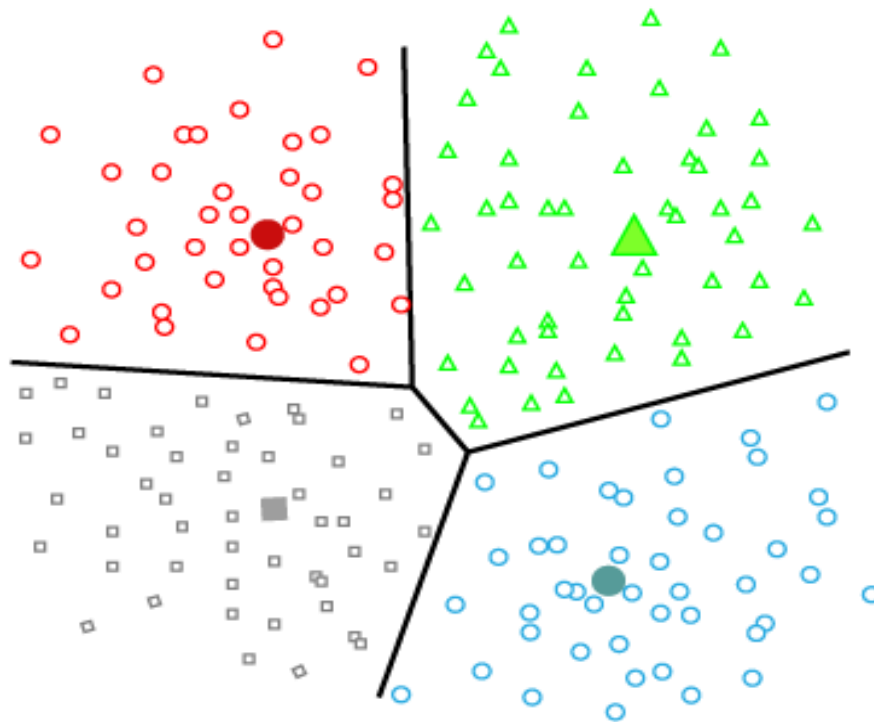
The clustering methods are broadly divided into **Hard clustering** (datapoint belongs to only one group) and **Soft Clustering** (data points can belong to another group also). But there are also other various approaches of Clustering exist. Below are the main clustering methods used in Machine learning:

1. **Partitioning Clustering**
2. **Density-Based Clustering**
3. **Distribution Model-Based Clustering**
4. **Hierarchical Clustering**
5. **Fuzzy Clustering**

## 1. Partitioning Clustering

It is a type of clustering that divides the data into non-hierarchical groups. It is also known as the **centroid-based method**. The most common example of partitioning clustering is the **K-Means Clustering algorithm**.

In this type, the dataset is divided into a set of  $k$  groups, where  $K$  is used to define the number of pre-defined groups. The cluster center is created in such a way that the distance between the data points of one cluster is minimum as compared to another cluster centroid.

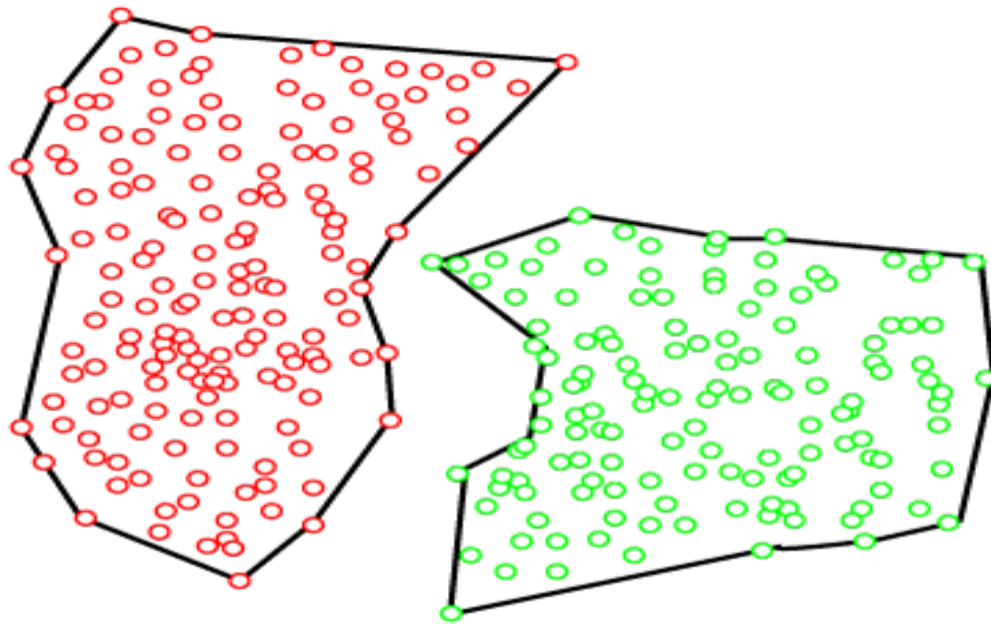


## 2. Density-Based Clustering

The density-based clustering method connects the highly-dense areas into clusters, and the arbitrarily shaped distributions are formed as long as the dense region can be connected. This algorithm does it by identifying different clusters in the dataset and connects the areas of high densities into clusters. The dense areas in data space are divided from each other by sparser areas.

These algorithms can face difficulty in clustering the data points if the dataset has varying densities and high dimensions.

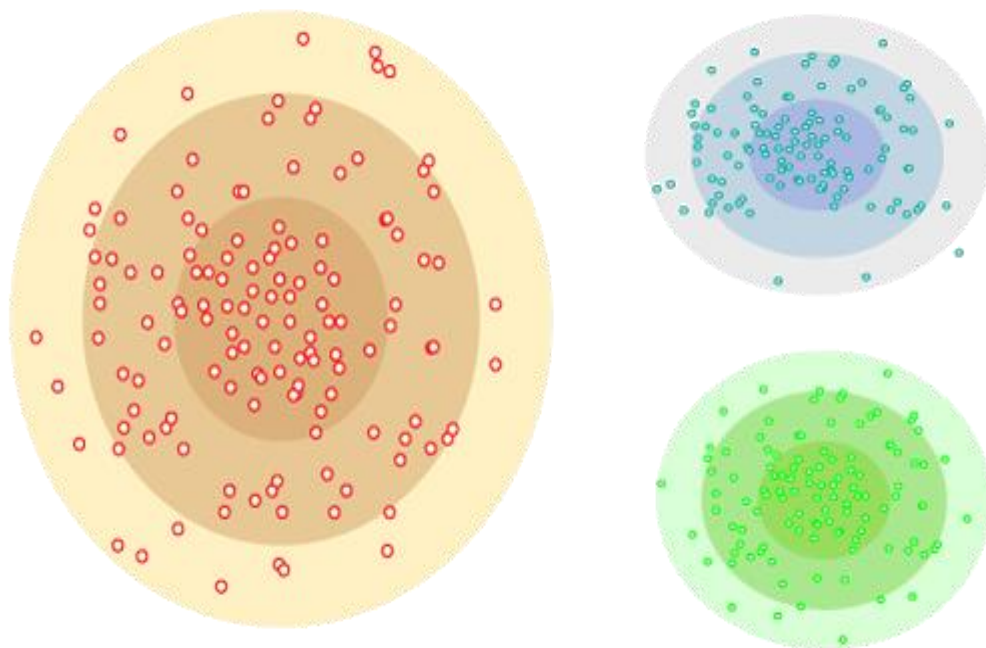




### 3. Distribution Model-Based Clustering

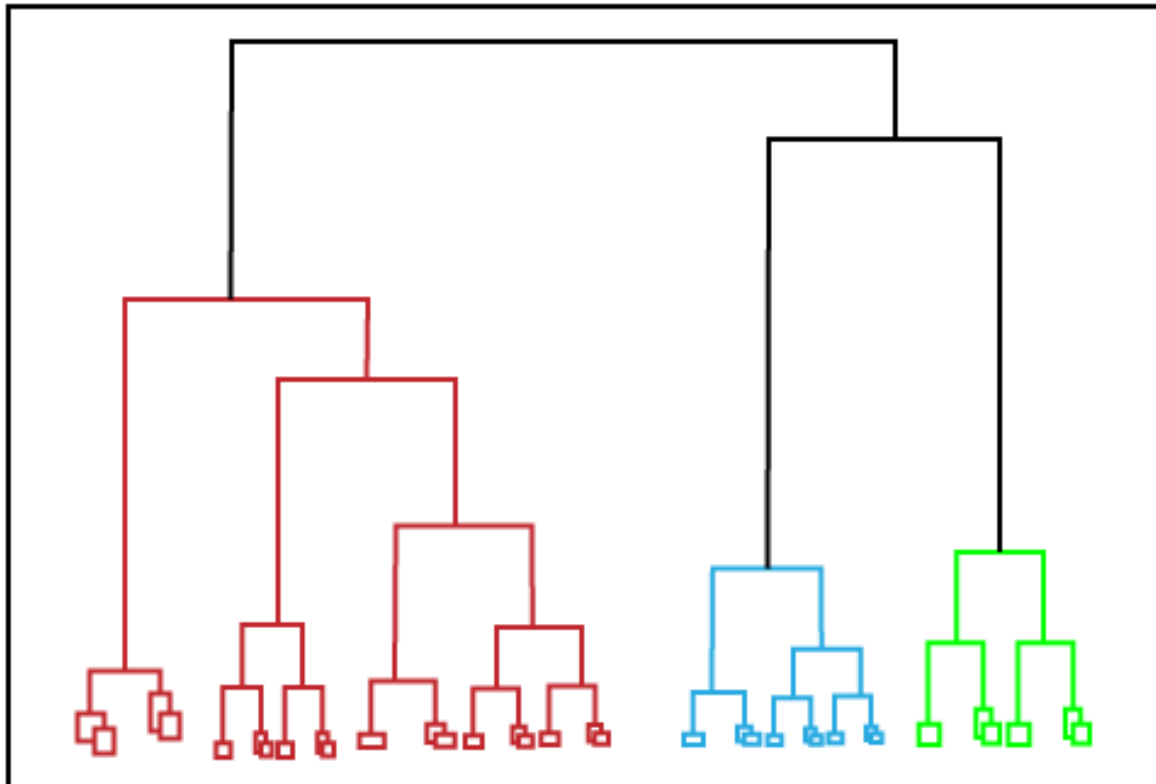
In the distribution model-based clustering method, the data is divided based on the probability of how a dataset belongs to a particular distribution. The grouping is done by assuming some distributions commonly **Gaussian Distribution**.

The example of this type is the **Expectation-Maximization Clustering algorithm** that uses Gaussian Mixture Models (GMM).



## 4. Hierarchical Clustering

Hierarchical clustering can be used as an alternative for the partitioned clustering as there is no requirement of pre-specifying the number of clusters to be created. In this technique, the dataset is divided into clusters to create a tree-like structure, which is also called a **dendrogram**. The observations or any number of clusters can be selected by cutting the tree at the correct level. The most common example of this method is the **Agglomerative Hierarchical algorithm**.



## 5. Fuzzy Clustering

Fuzzy clustering is a type of soft method in which a data object may belong to more than one group or cluster. Each dataset has a set of membership coefficients, which depend on the degree of membership to be in a cluster. **Fuzzy C-means algorithm** is the example of this type of clustering; it is sometimes also known as the Fuzzy k-means algorithm.

## Clustering Algorithms

The Clustering algorithms can be divided based on their models that are explained above. There are different types of clustering algorithms published, but only a few are commonly used. The clustering algorithm is based on the kind of data that we are using. Such as, some algorithms need to guess the number of clusters in the given dataset, whereas some are required to find the minimum distance between the observation of the dataset.

Here we are discussing mainly popular Clustering algorithms that are widely used in machine learning:

1. **K-Means algorithm:** The k-means algorithm is one of the most popular clustering algorithms. It classifies the dataset by dividing the samples into different clusters of equal variances. The number of clusters must be specified in this algorithm. It is fast with fewer computations required, with the linear complexity of  $O(n)$ .
2. **Mean-shift algorithm:** Mean-shift algorithm tries to find the dense areas in the smooth density of data points. It is an example of a centroid-based model, that works on updating the candidates for centroid to be the center of the points within a given region.
3. **DBSCAN Algorithm:** It stands for **Density-Based Spatial Clustering of Applications with Noise**. It is an example of a density-based model similar to the mean-shift, but with some remarkable advantages. In this algorithm, the areas of high density are separated by the areas of low density. Because of this, the clusters can be found in any arbitrary shape.
4. **Expectation-Maximization Clustering using GMM:** This algorithm can be used as an alternative for the k-means algorithm or for those cases where K-means can be failed. In GMM, it is assumed that the data points are Gaussian distributed.
5. **Agglomerative Hierarchical algorithm:** The Agglomerative hierarchical algorithm performs the bottom-up hierarchical clustering. In this, each data point is treated as a single cluster at the outset and then successively merged. The cluster hierarchy can be represented as a tree-structure.
6. **Affinity Propagation:** It is different from other clustering algorithms as it does not require to specify the number of clusters. In this, each data point sends a message between the pair of data points until convergence. It has  $O(N^2T)$  time complexity, which is the main drawback of this algorithm.

## Applications of Clustering

Below are some commonly known applications of clustering technique in Machine Learning:

- **In Identification of Cancer Cells:** The clustering algorithms are widely used for the identification of cancerous cells. It divides the cancerous and non-cancerous data sets into different groups.
- **In Search Engines:** Search engines also work on the clustering technique. The search result appears based on the closest object to the search query. It does it by grouping similar data objects in one group that is far from the other dissimilar objects. The accurate result of a query depends on the quality of the clustering algorithm used.
- **Customer Segmentation:** It is used in market research to segment the customers based on their choice and preferences.
- **In Biology:** It is used in the biology stream to classify different species of plants and animals using the image recognition technique.
- **In Land Use:** The clustering technique is used in identifying the area of similar lands use in the GIS database. This can be very useful to find that for what purpose the particular land should be used, that means for which purpose it is more suitable.

## Hierarchical Clustering in Machine Learning

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster and also known as **hierarchical cluster analysis** or HCA.

In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the **dendrogram**.

Sometimes the results of K-means clustering and hierarchical clustering may look similar, but they both differ depending on how they work. As there is no requirement to predetermine the number of clusters as we did in the K-Means algorithm.

The hierarchical clustering technique has two approaches:

1. **Agglomerative:** Agglomerative is a **bottom-up** approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.
2. **Divisive:** Divisive algorithm is the reverse of the agglomerative algorithm as it is a **top-down approach**.

## Why hierarchical clustering?

As we already have other clustering algorithms such as **K-Means Clustering**, then why we need hierarchical clustering? So, as we have seen in the K-means clustering that there are some challenges with this algorithm, which are a predetermined number of clusters, and it always tries to create the clusters of the same size. To solve these two challenges, we can opt for the hierarchical clustering algorithm because, in this algorithm, we don't need to have knowledge about the predefined number of clusters.

In this topic, we will discuss the Agglomerative Hierarchical clustering algorithm.

## Agglomerative Hierarchical clustering

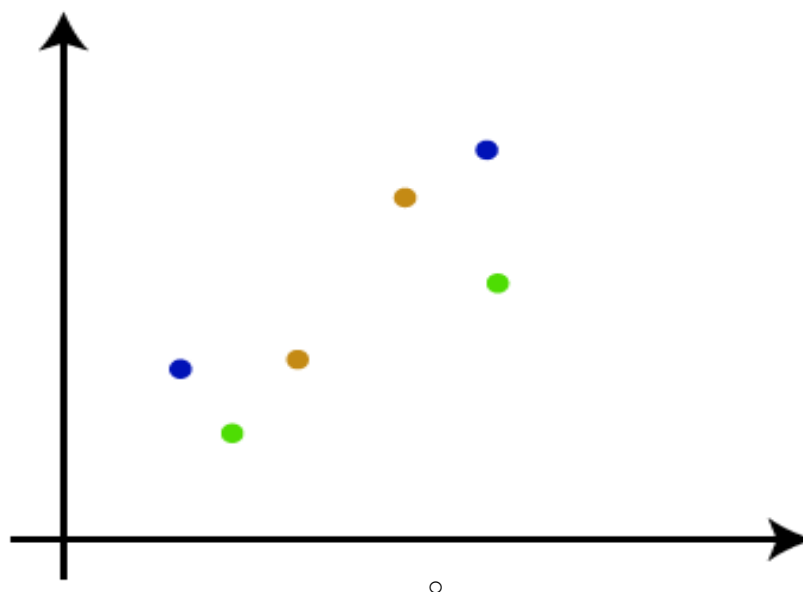
The agglomerative hierarchical clustering algorithm is a popular example of HCA. To group the datasets into clusters, it follows the **bottom-up approach**. It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together. It does this until all the clusters are merged into a single cluster that contains all the datasets.

This hierarchy of clusters is represented in the form of the dendrogram.

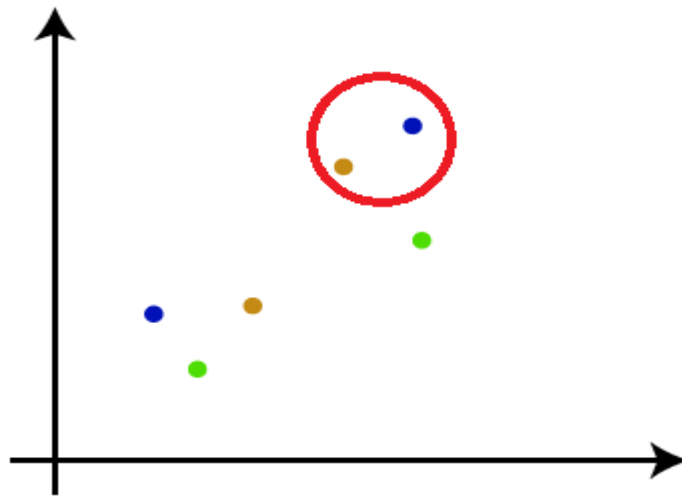
## How the Agglomerative Hierarchical clustering Work?

The working of the AHC algorithm can be explained using the below steps:

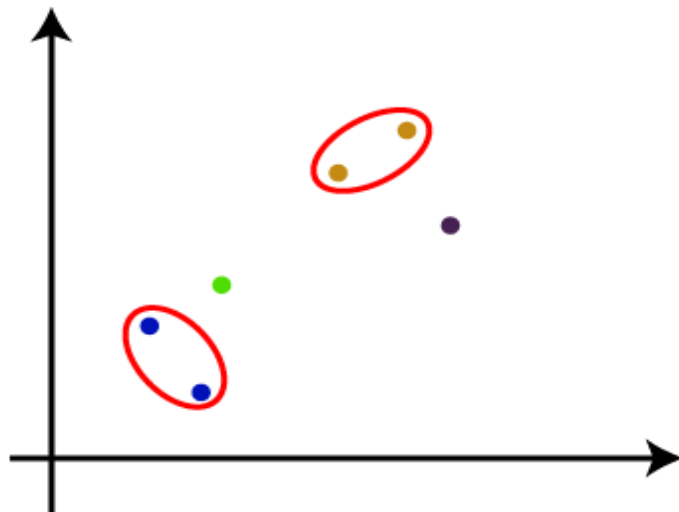
- **Step-1:** Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.



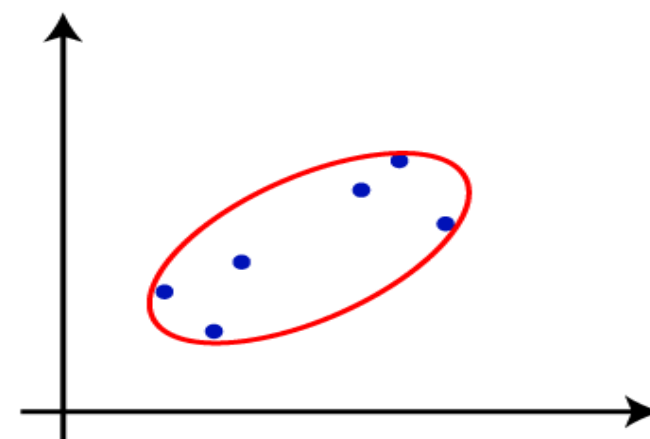
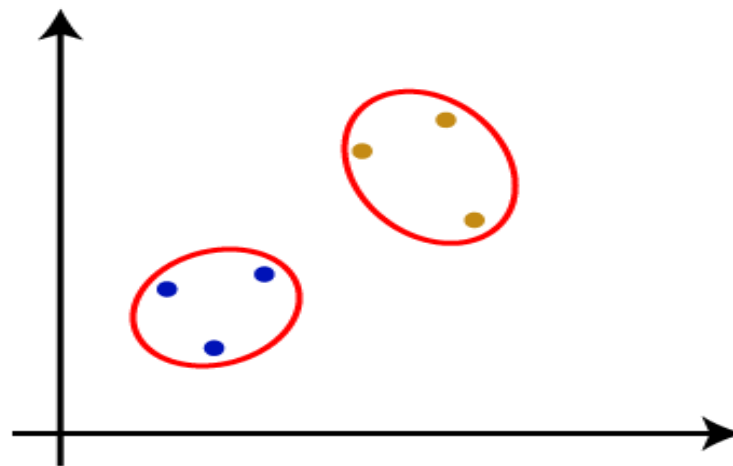
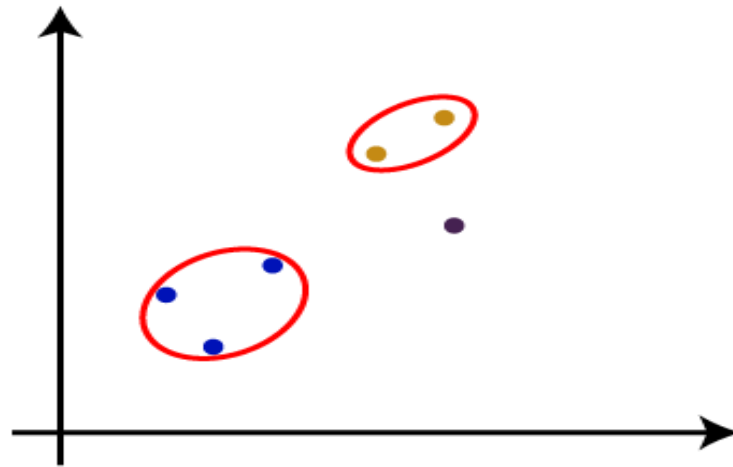
- **Step-2:** Take two closest data points or clusters and merge them to form one cluster. So, there will now be  $N-1$  clusters.



- **Step-3:** Again, take the two closest clusters and merge them together to form one cluster. There will be  $N-2$  clusters.



- **Step-4:** Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:



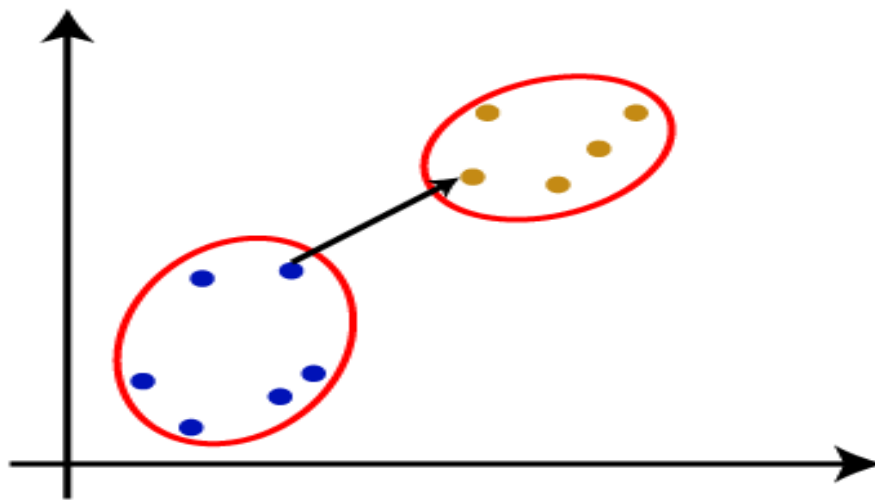
- **Step-5:** Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.

Note: To better understand hierarchical clustering, it is advised to have a look on k-means clustering

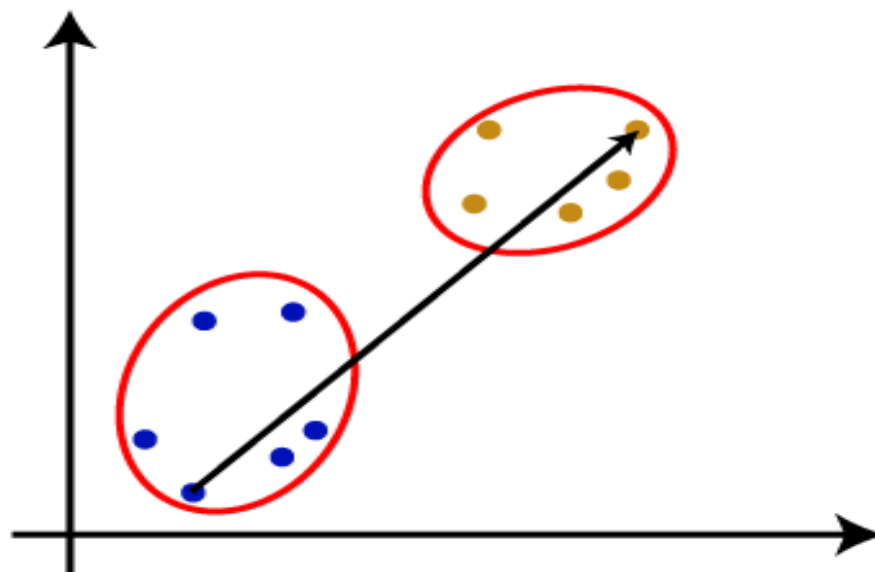
## Measure for the distance between two clusters

As we have seen, the **closest distance** between the two clusters is crucial for the hierarchical clustering. There are various ways to calculate the distance between two clusters, and these ways decide the rule for clustering. These measures are called **Linkage methods**. Some of the popular linkage methods are given below:

1. **Single Linkage:** It is the Shortest Distance between the closest points of the clusters. Consider the below image:

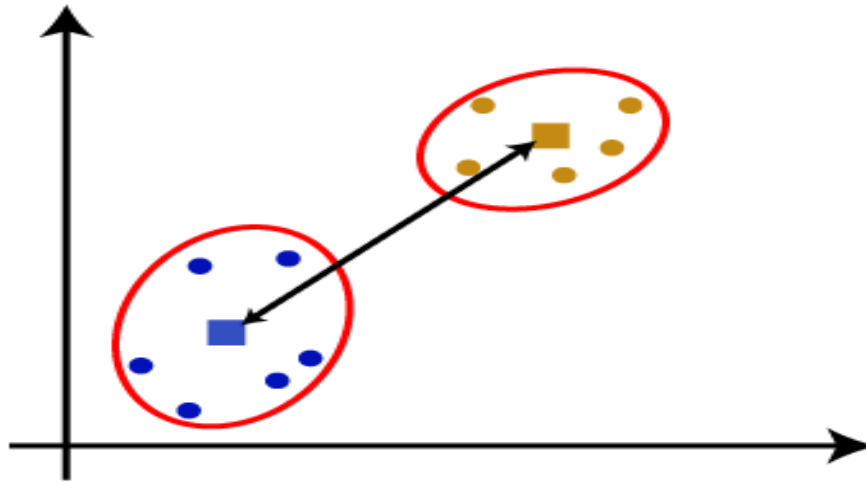


2. **Complete Linkage:** It is the farthest distance between the two points of two different clusters. It is one of the popular linkage methods as it forms tighter clusters than single-linkage.





3. **Average Linkage:** It is the linkage method in which the distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters. It is also one of the most popular linkage methods.
4. **Centroid Linkage:** It is the linkage method in which the distance between the centroid of the clusters is calculated. Consider the below image:

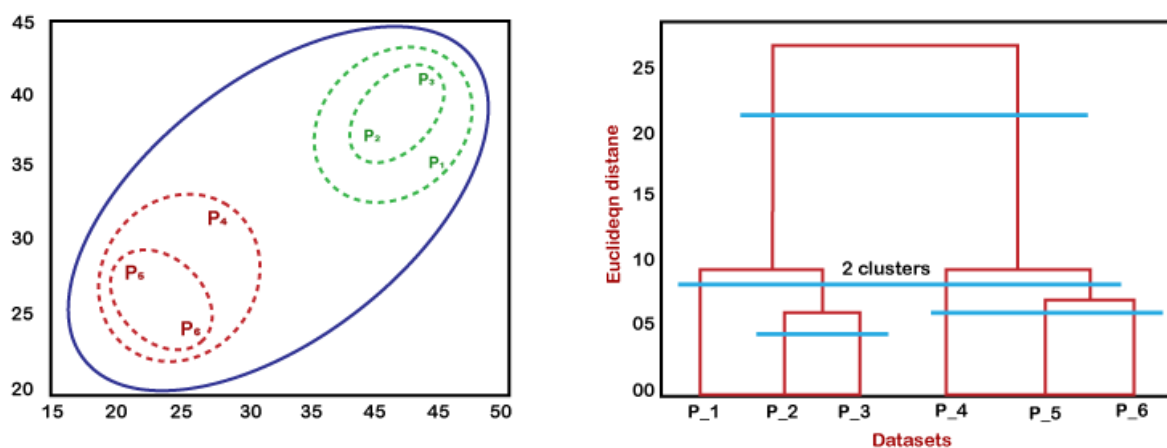


From the above-given approaches, we can apply any of them according to the type of problem or business requirement.

## Working of Dendrogram in Hierarchical clustering

The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.

The working of the dendrogram can be explained using the below diagram:



In the above diagram, the left part is showing how clusters are created in agglomerative clustering, and the right part is showing the corresponding dendrogram.

- As we have discussed above, firstly, the datapoints P2 and P3 combine together and form a cluster, correspondingly a dendrogram is created, which connects P2 and P3 with a rectangular shape. The height is decided according to the Euclidean distance between the data points.
- In the next step, P5 and P6 form a cluster, and the corresponding dendrogram is created. It is higher than of previous, as the Euclidean distance between P5 and P6 is a little bit greater than the P2 and P3.
- Again, two new dendrograms are created that combine P1, P2, and P3 in one dendrogram, and P4, P5, and P6, in another dendrogram.
- At last, the final dendrogram is created that combines all the data points together.

We can cut the dendrogram tree structure at any level as per our requirement.

## **K-Means Clustering Algorithm:**

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

### **What is K-Means Algorithm?**

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabelled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if  $K=2$ , there will be two clusters, and for  $K=3$ , there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabelled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabelled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

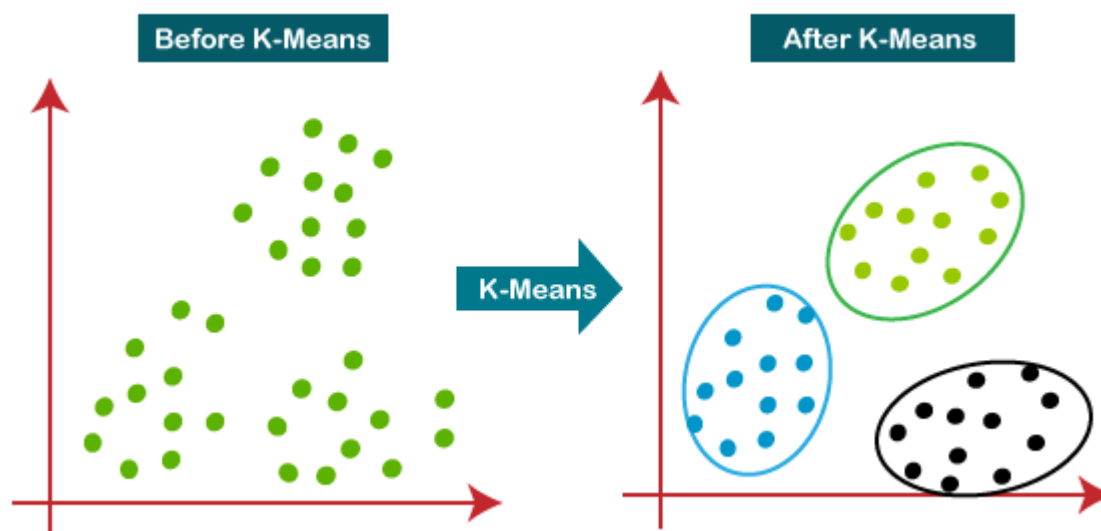
The algorithm takes the unlabelled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



## How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

**Step-1:** Select the number K to decide the number of clusters.

**Step-2:** Select random K points or centroids. (It can be other from the input dataset).

**Step-3:** Assign each data point to their closest centroid, which will form the predefined K clusters.

**Step-4:** Calculate the variance and place a new centroid of each cluster.

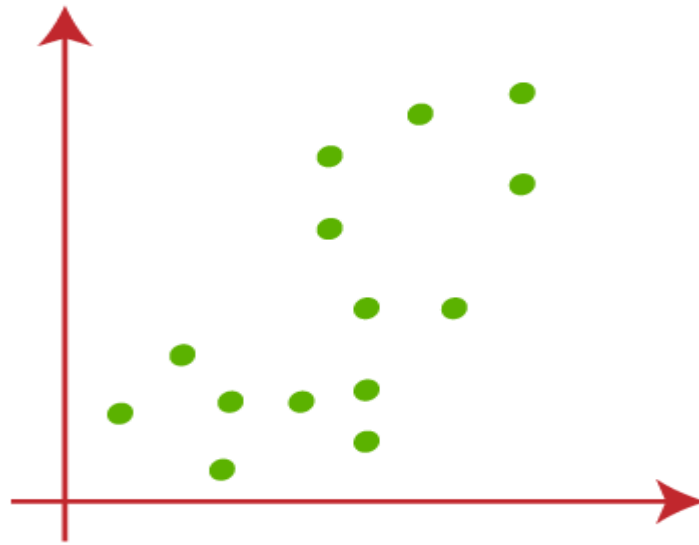
**Step-5:** Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

**Step-6:** If any reassignment occurs, then go to step-4 else go to FINISH.

**Step-7:** The model is ready.

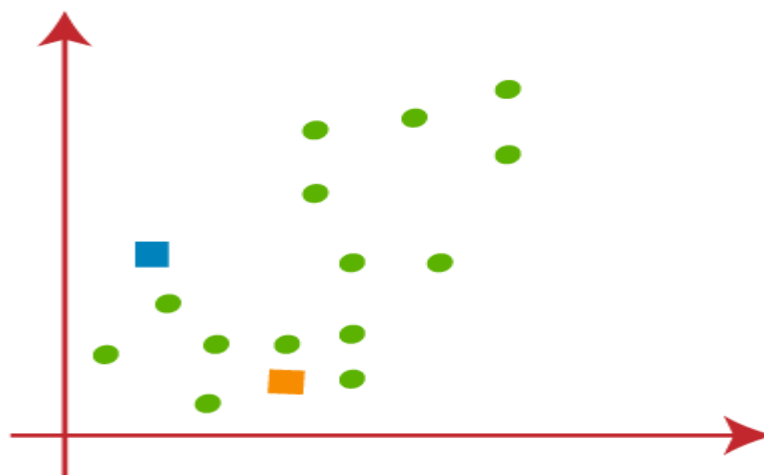
Let's understand the above steps by considering the visual plots:

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

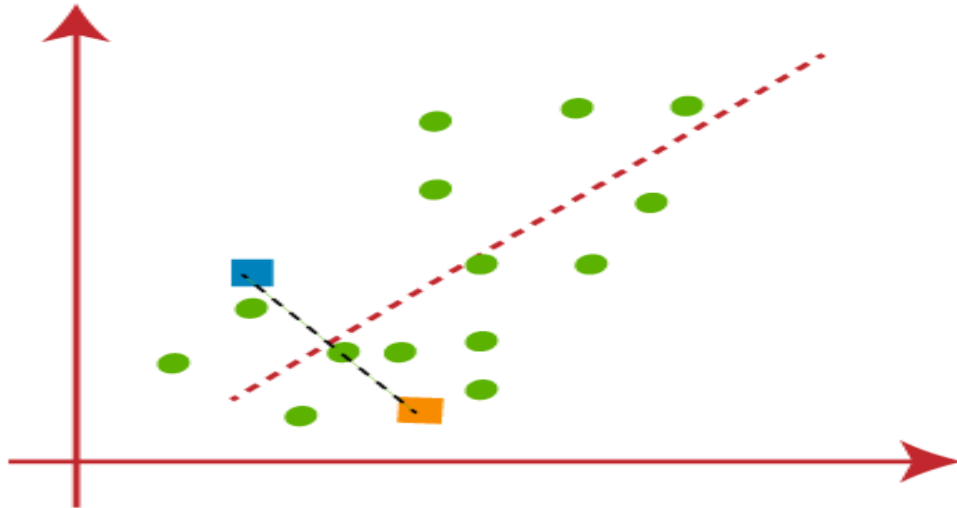


- Let's take number k of clusters, i.e.,  $K=2$ , to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.

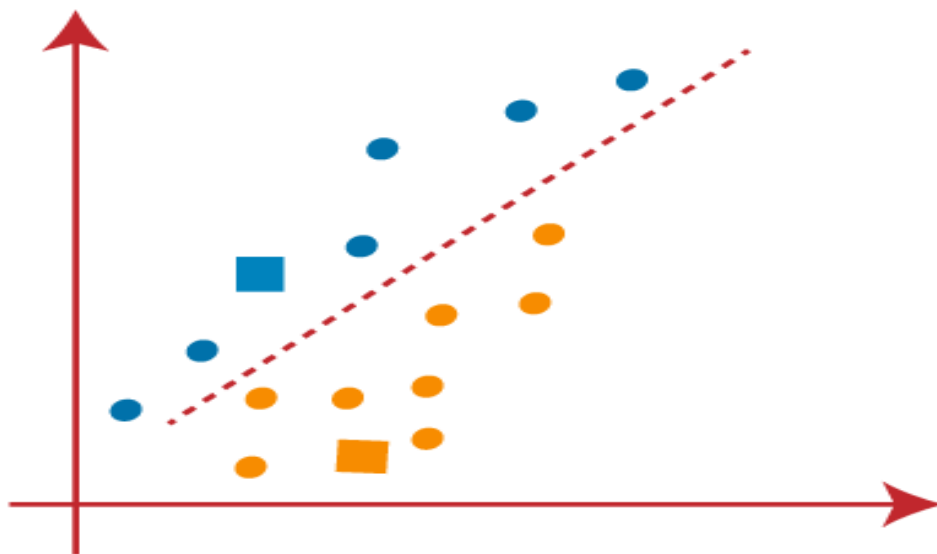
We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the below image:



- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids.  
Consider the below image:

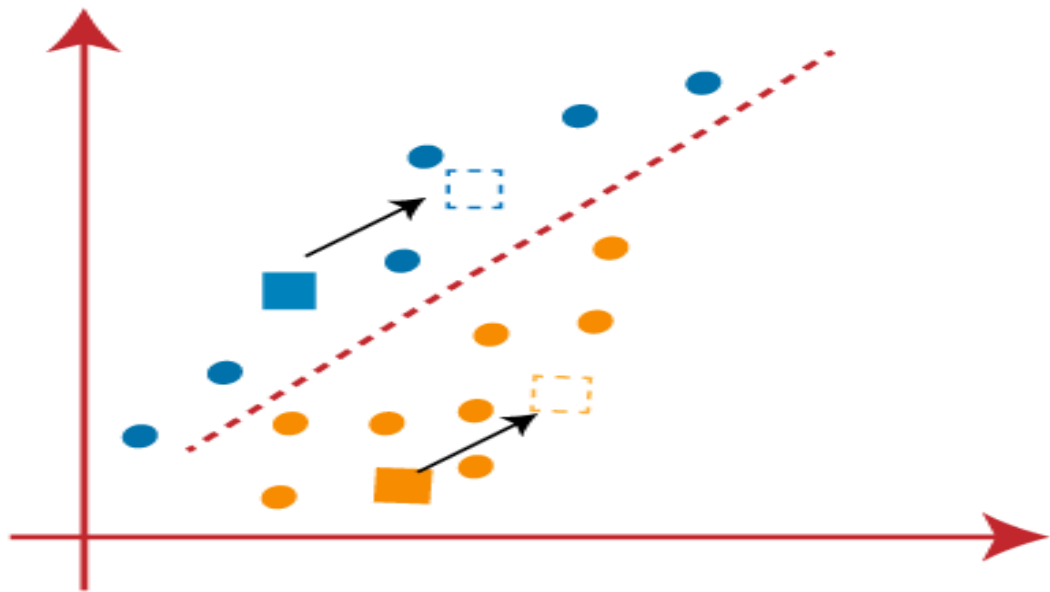


From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.

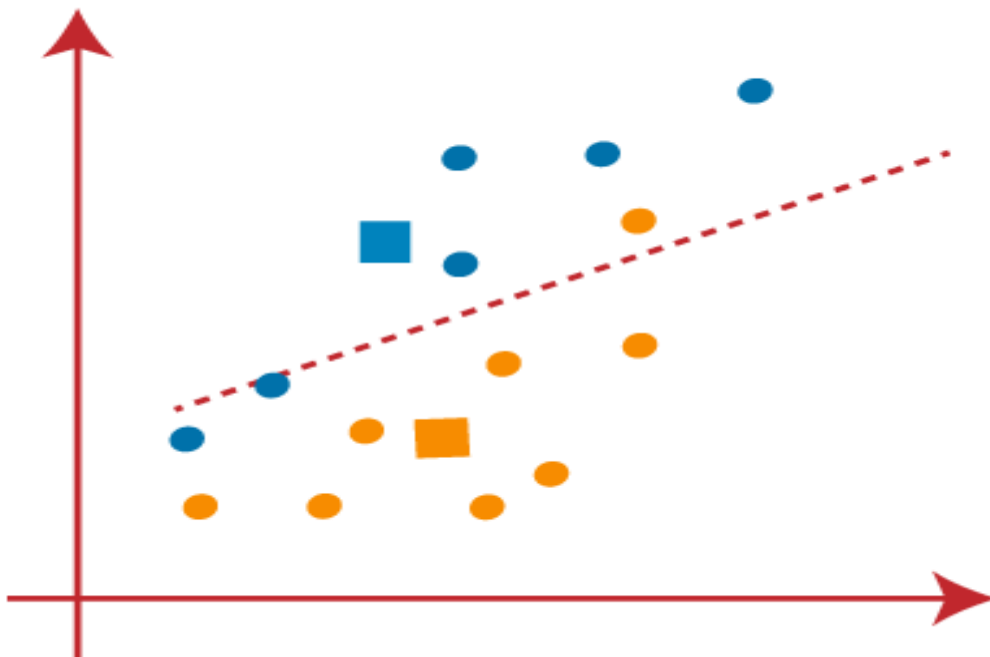


- As we need to find the closest cluster, so we will repeat the process by choosing a **new centroid**. To choose the new centroids, we will compute the

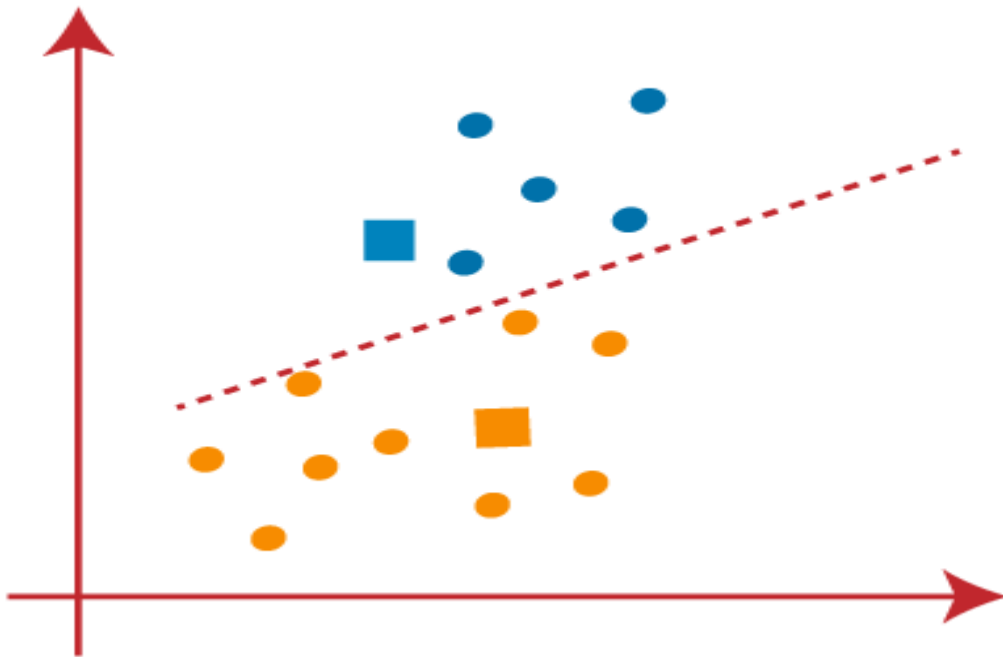
center of gravity of these centroids, and will find new centroids as below:



- Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:

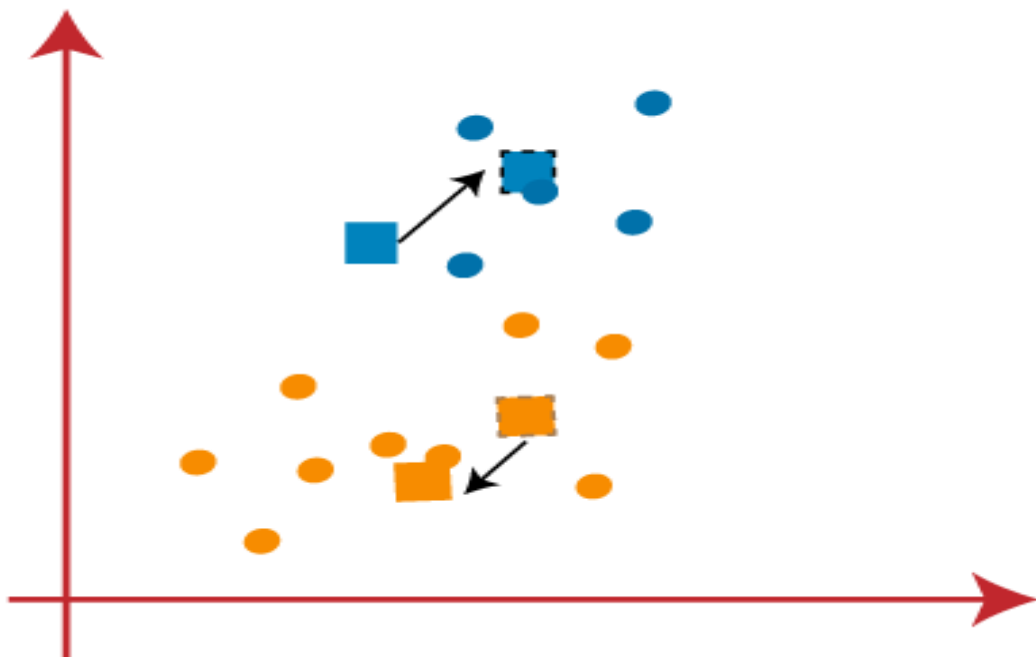


From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.

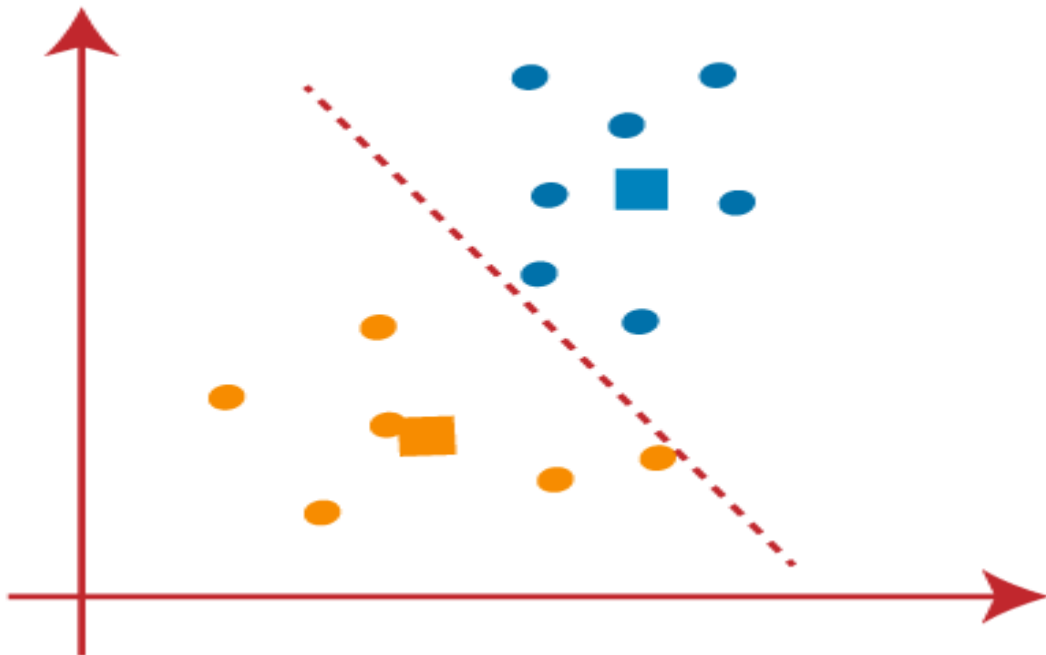


As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

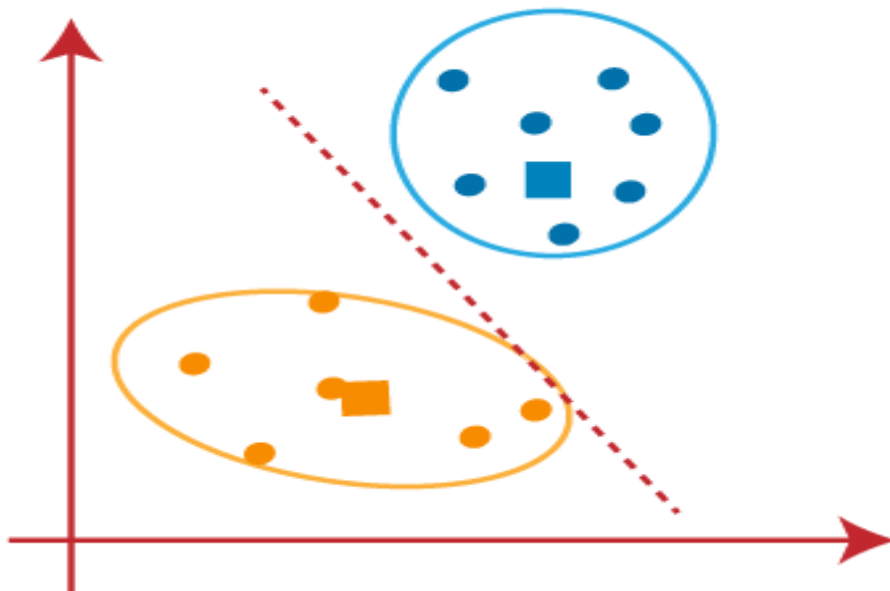
- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:



- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:

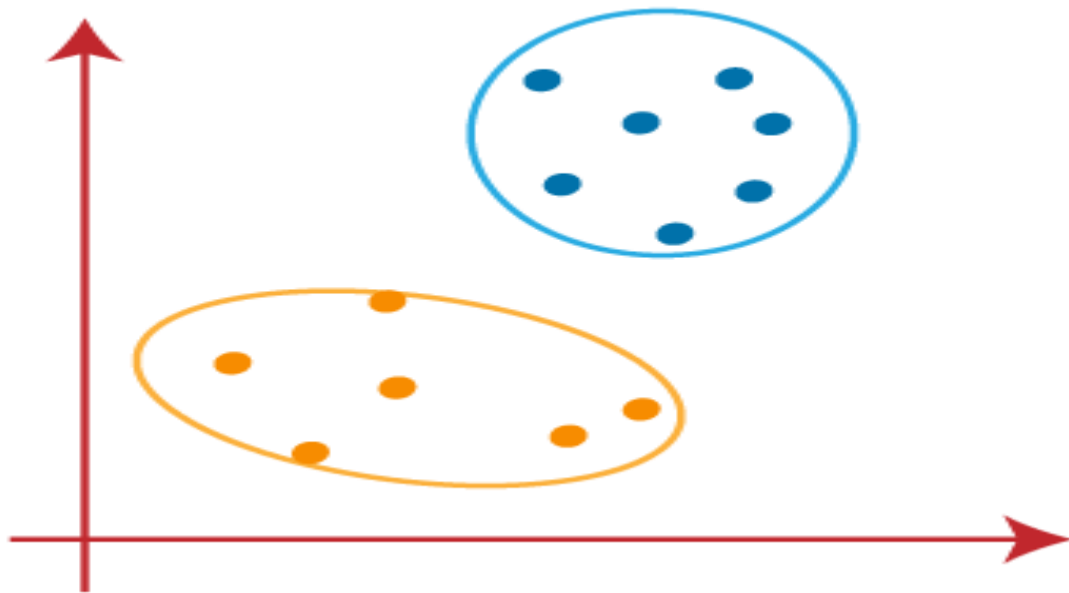


- We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:



As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:





## How to choose the value of "K number of clusters" in K-means Clustering?

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

### Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. **WCSS** stands for **Within Cluster Sum of Squares**, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$\text{WCSS} = \sum_{P_i \text{ in Cluster1}} \text{distance}(P_i C_1)^2 + \sum_{P_i \text{ in Cluster2}} \text{distance}(P_i C_2)^2 + \sum_{P_i \text{ in Cluster3}} \text{distance}(P_i C_3)^2$$

In the above formula of WCSS,

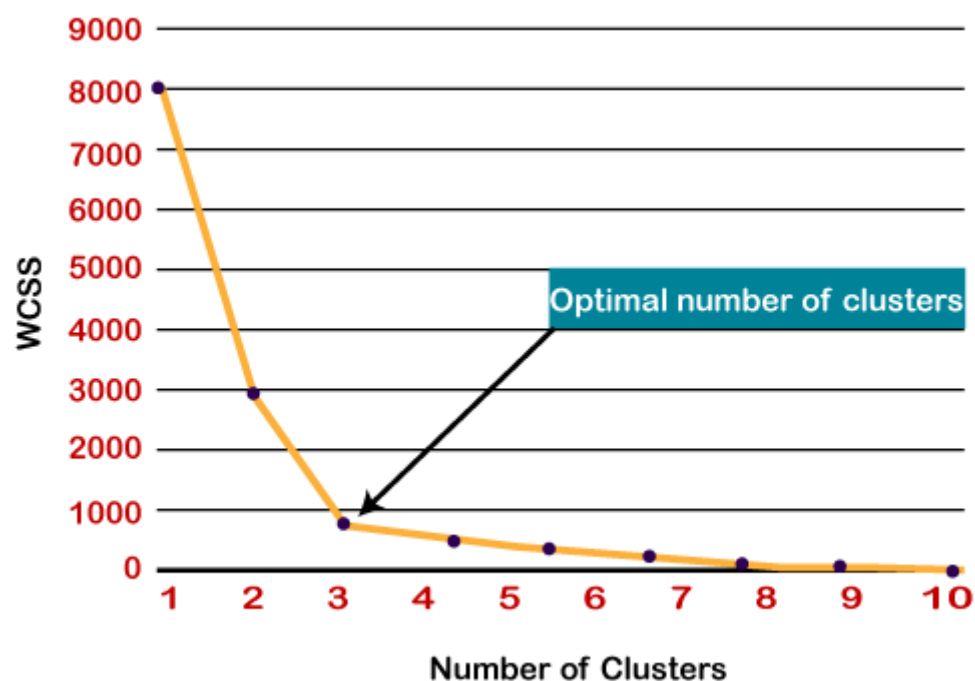
$\sum_{P_i \text{ in Cluster1}} \text{distance}(P_i C_1)^2$ : It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

- It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
- For each value of K, calculates the WCSS value.
- Plots a curve between calculated WCSS values and the number of clusters K.
- The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



Note: We can choose the number of clusters equal to the given data points. If we choose the number of clusters equal to the data points, then the value of WCSS becomes zero, and that will be the endpoint of the plot.

## Apriori Algorithm in Machine Learning

The Apriori algorithm uses frequent itemsets to generate association rules, and it is designed to work on the databases that contain transactions. With the help of these association rule, it determines how strongly or how weakly two objects are connected. This algorithm uses a **breadth-first search** and **Hash Tree** to calculate the itemset associations efficiently. It is the iterative process for finding the frequent itemsets from the large dataset.

This algorithm was given by the **R. Agrawal** and **Srikant** in the year **1994**. It is mainly used for *market basket analysis* and helps to find those products that can be bought together. It can also be used in the healthcare field to find drug reactions for patients.

## What is Frequent Itemset?

Frequent itemsets are those items whose support is greater than the threshold value or user-specified minimum support. It means if A & B are the frequent itemsets together, then individually A and B should also be the frequent itemset.

Suppose there are the two transactions: A= {1,2,3,4,5}, and B= {2,3,7}, in these two transactions, 2 and 3 are the frequent itemsets.

Note: To better understand the apriori algorithm, and related term such as support and confidence, it is recommended to understand the association rule learning.

## Steps for Apriori Algorithm

Below are the steps for the apriori algorithm:

**Step-1:** Determine the support of itemsets in the transactional database, and select the minimum support and confidence.

**Step-2:** Take all supports in the transaction with higher support value than the minimum or selected support value.

**Step-3:** Find all the rules of these subsets that have higher confidence value than the threshold or minimum confidence.

**Step-4:** Sort the rules as the decreasing order of lift.

## Apriori Algorithm Working

We will understand the apriori algorithm using an example and mathematical calculation:

**Example:** Suppose we have the following dataset that has various transactions, and from this dataset, we need to find the frequent itemsets and generate the association rules using the Apriori algorithm:

TID	ITEMSETS
T1	A, B
T2	B, D
T3	B, C
T4	A, B, D
T5	A, C
T6	B, C
T7	A, C
T8	A, B, C, E
T9	A, B, C

**Given: Minimum Support= 2, Minimum Confidence= 50%**

**Solution:**

### **Step-1: Calculating C1 and L1:**

- In the first step, we will create a table that contains support count (The frequency of each itemset individually in the dataset) of each itemset in the given dataset. This table is called the **Candidate set or C1**.

Itemset	Support_Count
A	6
B	7
C	5
D	2
E	1

- Now, we will take out all the itemsets that have the greater support count than the Minimum Support (2). It will give us the table for the **frequent itemset L1**. Since all the itemsets have greater or equal support count than the minimum support, except the E, so E itemset will be removed.

Itemset	Support_Count
A	6
B	7
C	5
D	2

### **Step-2: Candidate Generation C2, and L2:**

- In this step, we will generate C2 with the help of L1. In C2, we will create the pair of the itemsets of L1 in the form of subsets.

- After creating the subsets, we will again find the support count from the main transaction table of datasets, i.e., how many times these pairs have occurred together in the given dataset. So, we will get the below table for C2:

Itemset	Support_Count
{A, B}	4
{A,C}	4
{A, D}	1
{B, C}	4
{B, D}	2
{C, D}	0

- Again, we need to compare the C2 Support count with the minimum support count, and after comparing, the itemset with less support count will be eliminated from the table C2. It will give us the below table for L2

Itemset	Support_Count
{A, B}	4
{A, C}	4
{B, C}	4
{B, D}	2

**A, B, C, D**

### Step-3: Candidate generation C3, and L3:

- For C3, we will repeat the same two processes, but now we will form the C3 table with subsets of three itemsets together, and will calculate the support count from the dataset. It will give the below table:

Itemset	Support_Count
{A, B, C}	2
{B, C, D}	1
{A, C, D}	0
{A, B, D}	0

- Now we will create the L3 table. As we can see from the above C3 table, there is only one combination of itemset that has support count equal to the minimum support count. So, the L3 will have only one combination, i.e., **{A, B, C}**.

### Step-4: Finding the association rules for the subsets:

To generate the association rules, first, we will create a new table with the possible rules from the occurred combination {A, B,C}. For all the rules, we will calculate the

Confidence using formula  $\text{sup}(A \wedge B)/A$ . After calculating the confidence value for all rules, we will exclude the rules that have less confidence than the minimum threshold(50%).

**Consider the below table:**

Rules	Support	Confidence
$A \wedge B \rightarrow C$	2	$\text{Sup}\{(A \wedge B) \wedge C\}/\text{sup}(A \wedge B) = 2/4 = 0.5 = 50\%$
$B \wedge C \rightarrow A$	2	$\text{Sup}\{(B \wedge C) \wedge A\}/\text{sup}(B \wedge C) = 2/4 = 0.5 = 50\%$
$A \wedge C \rightarrow B$	2	$\text{Sup}\{(A \wedge C) \wedge B\}/\text{sup}(A \wedge C) = 2/4 = 0.5 = 50\%$
$C \rightarrow A \wedge B$	2	$\text{Sup}\{(C \wedge (A \wedge B))\}/\text{sup}(C) = 2/5 = 0.4 = 40\%$
$A \rightarrow B \wedge C$	2	$\text{Sup}\{(A \wedge (B \wedge C))\}/\text{sup}(A) = 2/6 = 0.33 = 33.33\%$
$B \rightarrow B \wedge C$	2	$\text{Sup}\{(B \wedge (B \wedge C))\}/\text{sup}(B) = 2/7 = 0.28 = 28\%$

As the given threshold or minimum confidence is 50%, so the first three rules  **$A \wedge B \rightarrow C$ ,  $B \wedge C \rightarrow A$ , and  $A \wedge C \rightarrow B$**  can be considered as the strong association rules for the given problem.

### Advantages of Apriori Algorithm

- This is easy to understand algorithm
- The join and prune steps of the algorithm can be easily implemented on large datasets.

### Disadvantages of Apriori Algorithm

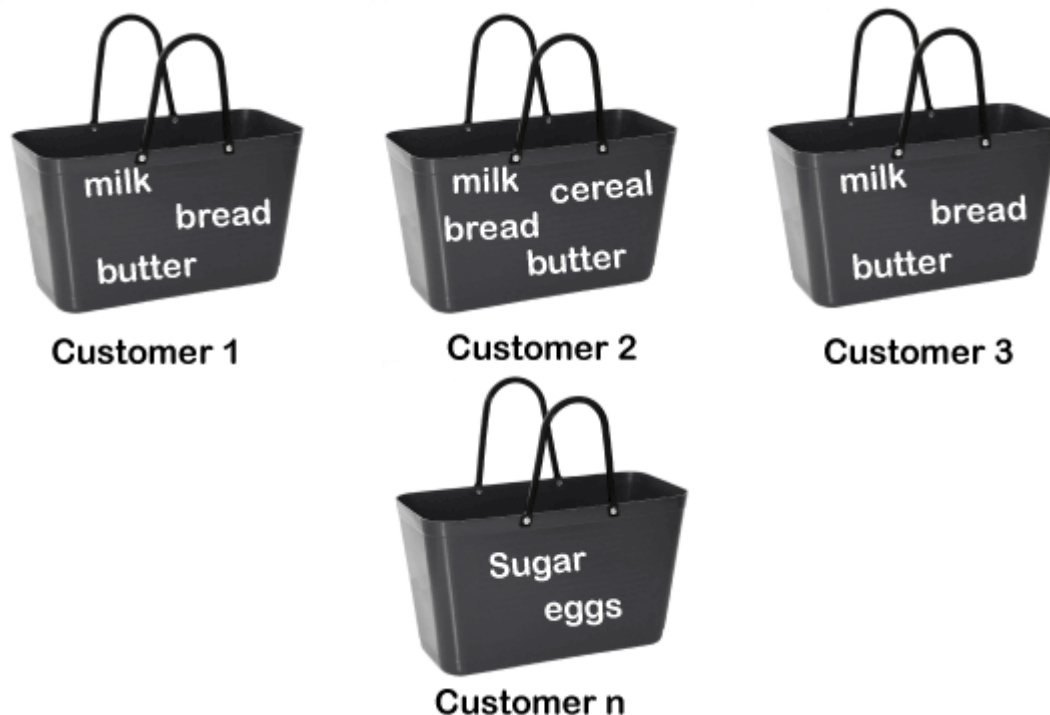
- The apriori algorithm works slow compared to other algorithms.
- The overall performance can be reduced as it scans the database for multiple times.
- The time complexity and space complexity of the apriori algorithm is  $O(2^D)$ , which is very high. Here D represents the horizontal width present in the database.

# ASSOCIATION RULES

Association rule learning is a type of unsupervised learning technique that checks for the dependency of one data item on another data item and maps accordingly so that it can be more profitable. It tries to find some interesting relations or associations among the variables of dataset. It is based on different rules to discover the interesting relations between variables in the database.

The association rule learning is one of the very important concepts of machine learning, and it is employed in **Market Basket analysis, Web usage mining, continuous production, etc.** Here market basket analysis is a technique used by the various big retailer to discover the associations between items. We can understand it by taking an example of a supermarket, as in a supermarket, all products that are purchased together are put together.

For example, if a customer buys bread, he most likely can also buy butter, eggs, or milk, so these products are stored within a shelf or mostly nearby. Consider the below diagram:



Association rule learning can be divided into three types of algorithms:

1. **Apriori**
2. **Eclat**
3. **F-P Growth Algorithm**

We will understand these algorithms in later chapters.

## How does Association Rule Learning work?

Association rule learning works on the concept of If and Else Statement, such as if A then B.



Here the If element is called **antecedent**, and then statement is called as **Consequent**. These types of relationships where we can find out some association or relation between two items is known as *single cardinality*. It is all about creating rules, and if the number of items increases, then cardinality also increases accordingly. So, to measure the associations between thousands of data items, there are several metrics. These metrics are given below:

- **Support**
- **Confidence**
- **Lift**

Let's understand each of them:

### Support

Support is the frequency of A or how frequently an item appears in the dataset. It is defined as the fraction of the transaction T that contains the itemset X. If there are X datasets, then for transactions T, it can be written as:

$$\text{Supp}(X) = \frac{\text{Freq}(X)}{T}$$

### Confidence

Confidence indicates how often the rule has been found to be true. Or how often the items X and Y occur together in the dataset when the occurrence of X is already given. It is the ratio of the transaction that contains X and Y to the number of records that contain X.

$$\text{Confidence} = \frac{\text{Freq}(X,Y)}{\text{Freq}(X)}$$



## Lift

It is the strength of any rule, which can be defined as below formula:

$$\text{Lift} = \frac{\text{Supp}(X,Y)}{\text{Supp}(X) \times \text{Supp}(Y)}$$

It is the ratio of the observed support measure and expected support if X and Y are independent of each other. It has three possible values:

- If **Lift= 1**: The probability of occurrence of antecedent and consequent is independent of each other.
- **Lift>1**: It determines the degree to which the two itemsets are dependent to each other.
- **Lift<1**: It tells us that one item is a substitute for other items, which means one item has a negative effect on another.

## Types of Association Rule Learning

Association rule learning can be divided into three algorithms:

### Apriori Algorithm

This algorithm uses frequent datasets to generate association rules. It is designed to work on the databases that contain transactions. This algorithm uses a breadth-first search and Hash Tree to calculate the itemset efficiently.

It is mainly used for market basket analysis and helps to understand the products that can be bought together. It can also be used in the healthcare field to find drug reactions for patients.

### Eclat Algorithm

Eclat algorithm stands for **Equivalence Class Transformation**. This algorithm uses a depth-first search technique to find frequent itemsets in a transaction database. It performs faster execution than Apriori Algorithm.

### F-P Growth Algorithm

The F-P growth algorithm stands for **Frequent Pattern**, and it is the improved version of the Apriori Algorithm. It represents the database in the form of a tree structure that is known as a frequent pattern or tree. The purpose of this frequent tree is to extract the most frequent patterns.

## Applications of Association Rule Learning

It has various applications in machine learning and data mining. Below are some popular applications of association rule learning:

- **Market Basket Analysis:** It is one of the popular examples and applications of association rule mining. This technique is commonly used by big retailers to determine the association between items.
- **Medical Diagnosis:** With the help of association rules, patients can be cured easily, as it helps in identifying the probability of illness for a particular disease.
- **Protein Sequence:** The association rules help in determining the synthesis of artificial Proteins.
- It is also used for the **Catalog Design** and **Loss-leader Analysis** and many more other applications.

# CONFUSION MATRIX

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an **error matrix**. Some features of Confusion matrix are given below:

- For the 2 prediction classes of classifiers, the matrix is of 2\*2 table, for 3 classes, it is 3\*3 table, and so on.
- The matrix is divided into two dimensions, that are **predicted values** and **actual values** along with the total number of predictions.
- Predicted values are those values, which are predicted by the model, and actual values are the true values for the given observations.
- It looks like the below table:

n = total predictions	Actual: No	Actual: Yes
Predicted: No	True Negative	False Positive
Predicted: Yes	False Negative	True Positive

The above table has the following cases:

- **True Negative:** Model has given prediction No, and the real or actual value was also No.
- **True Positive:** The model has predicted yes, and the actual value was also true.
- **False Negative:** The model has predicted no, but the actual value was Yes, it is also called as **Type-II error**.
- **False Positive:** The model has predicted Yes, but the actual value was No. It is also called a **Type-I error**.

## Need for Confusion Matrix in Machine learning

- It evaluates the performance of the classification models, when they make predictions on test data, and tells how good our classification model is.
- It not only tells the error made by the classifiers but also the type of errors such as it is either type-I or type-II error.

- With the help of the confusion matrix, we can calculate the different parameters for the model, such as accuracy, precision, etc.

**Example:** We can understand the confusion matrix using an example.

Suppose we are trying to create a model that can predict the result for the disease that is either a person has that disease or not. So, the confusion matrix for this is given as:

n = 100	Actual: No	Actual: Yes	
Predicted: No	TN: 65	FP: 3	68
Predicted: Yes	FN: 8	TP: 24	32
	73	27	

From the above example, we can conclude that:

- The table is given for the two-class classifier, which has two predictions "Yes" and "NO." Here, Yes defines that patient has the disease, and No defines that patient does not has that disease.
- The classifier has made a total of **100 predictions**. Out of 100 predictions, **89 are true predictions**, and **11 are incorrect predictions**.
- The model has given prediction "yes" for 32 times, and "No" for 68 times. Whereas the actual "Yes" was 27, and actual "No" was 73 times.

### Calculations using Confusion Matrix:

We can perform various calculations for the model, such as the model's accuracy, using this matrix. These calculations are given below:

- **Classification Accuracy:** It is one of the important parameters to determine the accuracy of the classification problems. It defines how often the model predicts the correct output. It can be calculated as the ratio of the number of correct predictions made by the classifier to all number of predictions made by the classifiers. The formula is given below:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN}$$

- **Misclassification rate:** It is also termed as Error rate, and it defines how often the model gives the wrong predictions. The value of error rate can be calculated

as the number of incorrect predictions to all number of the predictions made by the classifier. The formula is given below:

$$\text{Error rate} = \frac{FP + FN}{TP + FP + FN + TN}$$

- **Precision:** It can be defined as the number of correct outputs provided by the model or out of all positive classes that have predicted correctly by the model, how many of them were actually true. It can be calculated using the below formula:

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall:** It is defined as the out of total positive classes, how our model predicted correctly. The recall must be as high as possible.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F-measure:** If two models have low precision and high recall or vice versa, it is difficult to compare these models. So, for this purpose, we can use F-score. This score helps us to evaluate the recall and precision at the same time. The F-score is maximum if the recall is equal to the precision. It can be calculated using the below formula:

$$\text{F-measure} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$$

Other important terms used in Confusion Matrix:

- **Null Error rate:** It defines how often our model would be incorrect if it always predicted the majority class. As per the accuracy paradox, it is said that "*the best classifier has a higher error rate than the null error rate.*"
- **ROC Curve:** The ROC is a graph displaying a classifier's performance for all possible thresholds. The graph is plotted between the true positive rate (on the Y-axis) and the false Positive rate (on the x-axis).

# CROSS-VALIDATION

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. ***We can also say that it is a technique to check how a statistical model generalizes to an independent dataset.***

In machine learning, there is always the need to test the stability of the model. It means based only on the training dataset; we can't fit our model on the training dataset. For this purpose, we reserve a particular sample of the dataset, which was not part of the training dataset. After that, we test our model on that sample before deployment, and this complete process comes under cross-validation. This is something different from the general train-test split.

Hence the basic steps of cross-validations are:

- Reserve a subset of the dataset as a validation set.
- Provide the training to the model using the training dataset.
- Now, evaluate model performance using the validation set. If the model performs well with the validation set, perform the further step, else check for the issues.

## Methods used for Cross-Validation

There are some common methods that are used for cross-validation. These methods are given below:

1. **Validation Set Approach**
2. **Leave-P-out cross-validation**
3. **Leave one out cross-validation**
4. **K-fold cross-validation**
5. **Stratified k-fold cross-validation**

## Validation Set Approach

We divide our input dataset into a training set and test or validation set in the validation set approach. Both the subsets are given 50% of the dataset.

But it has one of the big disadvantages that we are just using a 50% dataset to train our model, so the model may miss out to capture important information of the dataset. It also tends to give the underfitted model.

## 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:

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

## K-Fold Cross-Validation

K-fold cross-validation approach divides the input dataset into  $K$  groups of samples of equal sizes. These samples are called **folds**. For each learning set, the prediction function uses  $k-1$  folds, and the rest of the folds are used for the test set. This approach is a very popular CV approach because it is easy to understand, and the output is less biased than other methods.

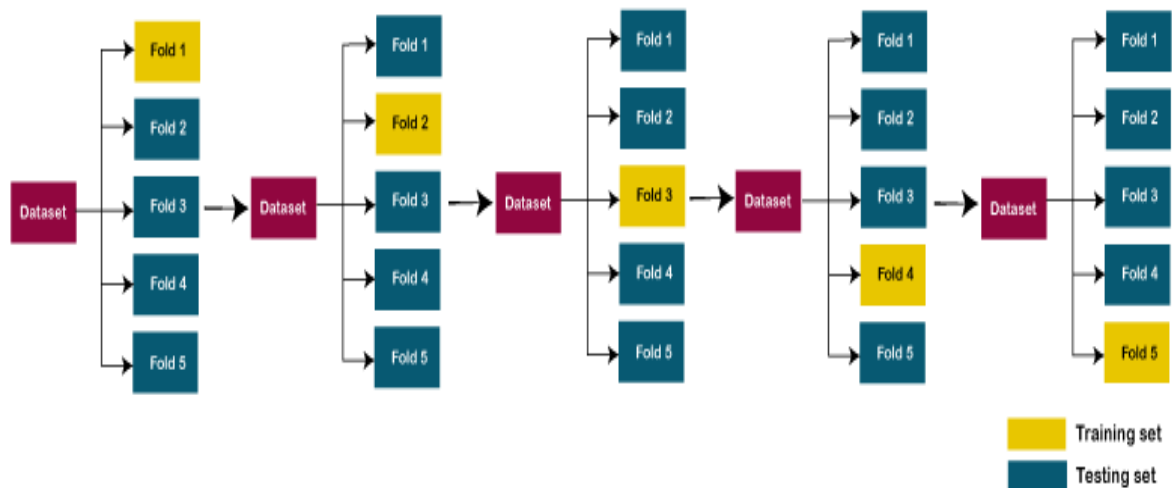
The steps for  $k$ -fold cross-validation are:

- Split the input dataset into  $K$  groups
- For each group:
  - Take one group as the reserve or test data set.
  - Use remaining groups as the training dataset
  - Fit the model on the training set and evaluate the performance of the model using the test set.

Let's take an example of 5-folds cross-validation. So, the dataset is grouped into 5 folds. On 1<sup>st</sup> iteration, the first fold is reserved for test the model, and rest are used to

train the model. On 2<sup>nd</sup> iteration, the second fold is used to test the model, and rest are used to train the model. This process will continue until each fold is not used for the test fold.

Consider the below diagram:



## Stratified k-fold cross-validation

This technique is similar to k-fold cross-validation with some little changes. This approach works on stratification concept, it is a process of rearranging the data to ensure that each fold or group is a good representative of the complete dataset. To deal with the bias and variance, it is one of the best approaches.

It can be understood with an example of housing prices, such that the price of some houses can be much high than other houses. To tackle such situations, a stratified k-fold cross-validation technique is useful.

## 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.



## Comparison of Cross-validation to train/test split in Machine Learning

- **Train/test split:** The input data is divided into two parts, that are training set and test set on a ratio of 70:30, 80:20, etc. It provides a high variance, which is one of the biggest disadvantages.
  - **Training Data:** The training data is used to train the model, and the dependent variable is known.
  - **Test Data:** The test data is used to make the predictions from the model that is already trained on the training data. This has the same features as training data but not the part of that.
- **Cross-Validation dataset:** It is used to overcome the disadvantage of train/test split by splitting the dataset into groups of train/test splits, and averaging the result. It can be used if we want to optimize our model that has been trained on the training dataset for the best performance. It is more efficient as compared to train/test split as every observation is used for the training and testing both.

## Limitations of Cross-Validation

There are some limitations of the cross-validation technique, which are given below:

- For the ideal conditions, it provides the optimum output. But for the inconsistent data, it may produce a drastic result. So, it is one of the big disadvantages of cross-validation, as there is no certainty of the type of data in machine learning.
- In predictive modeling, the data evolves over a period, due to which, it may face the differences between the training set and validation sets. Such as if we create a model for the prediction of stock market values, and the data is trained on the previous 5 years stock values, but the realistic future values for the next 5 years may drastically different, so it is difficult to expect the correct output for such situations.

## Applications of Cross-Validation

- This technique can be used to compare the performance of different predictive modeling methods.
- It has great scope in the medical research field.
- It can also be used for the meta-analysis, as it is already being used by the data scientists in the field of medical statistics.

# DIMENSIONALITY REDUCTION TECHNIQUES

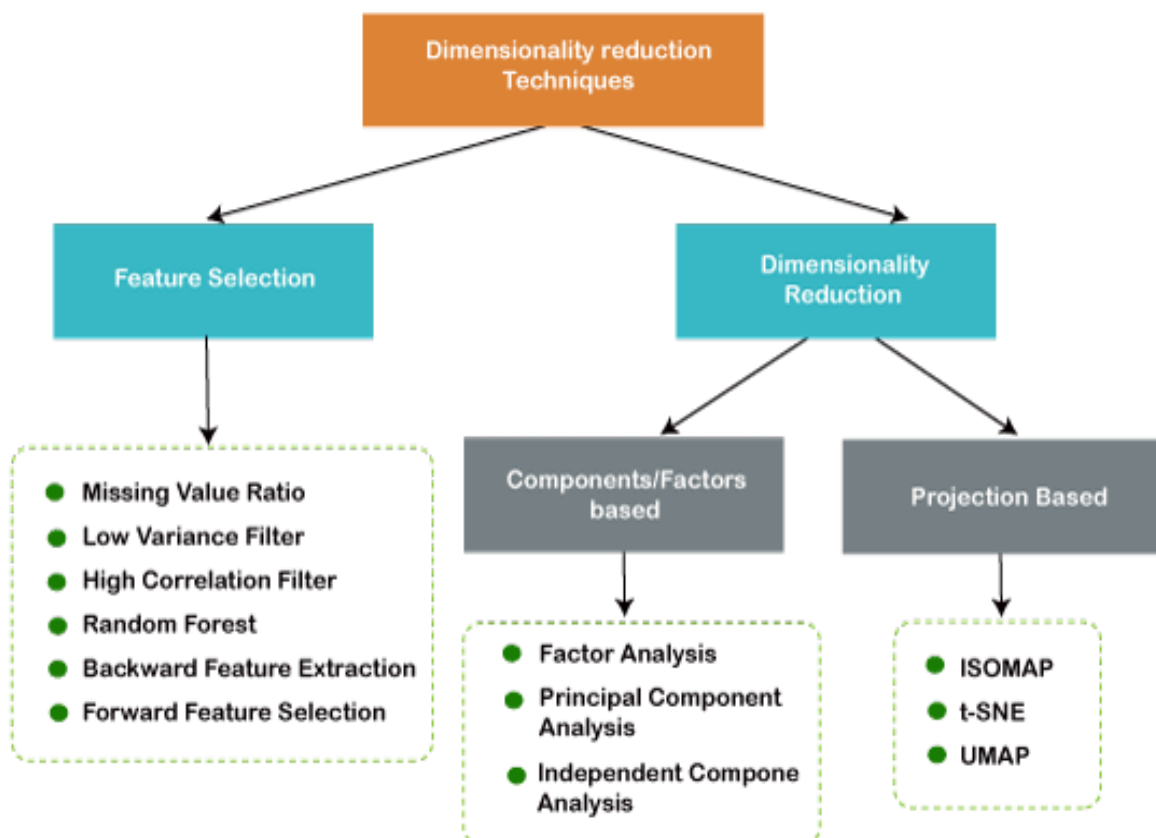
## What is Dimensionality Reduction?

The number of input features, variables, or columns present in a given dataset is known as dimensionality, and the process to reduce these features is called dimensionality reduction.

A dataset contains a huge number of input features in various cases, which makes the predictive modeling task more complicated. Because it is very difficult to visualize or make predictions for the training dataset with a high number of features, for such cases, dimensionality reduction techniques are required to use.

Dimensionality reduction technique can be defined as, ***"It is a way of converting the higher dimensions dataset into lesser dimensions dataset ensuring that it provides similar information."*** These techniques are widely used in machine learning for obtaining a better fit predictive model while solving the classification and regression problems.

It is commonly used in the fields that deal with high-dimensional data, such as **speech recognition, signal processing, bioinformatics, etc.** It can also be used for **data visualization, noise reduction, cluster analysis, etc.**



## **The Curse of Dimensionality**

Handling the high-dimensional data is very difficult in practice, commonly known as the *curse of dimensionality*. If the dimensionality of the input dataset increases, any machine learning algorithm and model becomes more complex. As the number of features increases, the number of samples also gets increased proportionally, and the chance of overfitting also increases. If the machine learning model is trained on high-dimensional data, it becomes overfitted and results in poor performance.

Hence, it is often required to reduce the number of features, which can be done with dimensionality reduction.

## **Benefits of applying Dimensionality Reduction**

Some benefits of applying dimensionality reduction technique to the given dataset are given below:

- By reducing the dimensions of the features, the space required to store the dataset also gets reduced.
- Less Computation training time is required for reduced dimensions of features.
- Reduced dimensions of features of the dataset help in visualizing the data quickly.
- It removes the redundant features (if present) by taking care of multicollinearity.

## **Disadvantages of dimensionality Reduction**

There are also some disadvantages of applying the dimensionality reduction, which are given below:

- Some data may be lost due to dimensionality reduction.
- In the PCA dimensionality reduction technique, sometimes the principal components required to consider are unknown.

## **Approaches of Dimension Reduction**

There are two ways to apply the dimension reduction technique, which are given below:

### **Feature Selection**

Feature selection is the process of selecting the subset of the relevant features and leaving out the irrelevant features present in a dataset to build a model of high

accuracy. In other words, it is a way of selecting the optimal features from the input dataset.

Three methods are used for the feature selection:

## **1. Filters Methods**

In this method, the dataset is filtered, and a subset that contains only the relevant features is taken. Some common techniques of filters method are:

- **Correlation**
- **Chi-Square Test**
- **ANOVA**
- **Information Gain, etc.**

## **2. Wrappers Methods**

The wrapper method has the same goal as the filter method, but it takes a machine learning model for its evaluation. In this method, some features are fed to the ML model, and evaluate the performance. The performance decides whether to add those features or remove to increase the accuracy of the model. This method is more accurate than the filtering method but complex to work. Some common techniques of wrapper methods are:

- Forward Selection
- Backward Selection
- Bi-directional Elimination

## **3. Embedded Methods:**

Embedded methods check the different training iterations of the machine learning model and evaluate the importance of each feature. Some common techniques of Embedded methods are:

- **LASSO**
- **Elastic Net**
- **Ridge Regression, etc.**

## **Feature Extraction:**

Feature extraction is the process of transforming the space containing many dimensions into space with fewer dimensions. This approach is useful when we want

to keep the whole information but use fewer resources while processing the information.

Some common feature extraction techniques are:

- a. Principal Component Analysis
- b. Linear Discriminant Analysis
- c. Kernel PCA
- d. Quadratic Discriminant Analysis

## Common techniques of Dimensionality Reduction

- a. **Principal Component Analysis**
- b. **Backward Elimination**
- c. **Forward Selection**
- d. **Score comparison**
- e. **Missing Value Ratio**
- f. **Low Variance Filter**
- g. **High Correlation Filter**
- h. **Random Forest**
- i. **Factor Analysis**
- j. **Auto-Encoder**

## Principal Component Analysis (PCA)

Principal Component Analysis is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. These new transformed features are called the **Principal Components**. It is one of the popular tools that is used for exploratory data analysis and predictive modeling.

PCA works by considering the variance of each attribute because the high attribute shows the good split between the classes, and hence it reduces the dimensionality. Some real-world applications of PCA are ***image processing, movie recommendation system, optimizing the power allocation in various communication channels.***

## Backward Feature Elimination

The backward feature elimination technique is mainly used while developing Linear Regression or Logistic Regression model. Below steps are performed in this technique to reduce the dimensionality or in feature selection:

- In this technique, firstly, all the  $n$  variables of the given dataset are taken to train the model.
- The performance of the model is checked.
- Now we will remove one feature each time and train the model on  $n-1$  features for  $n$  times, and will compute the performance of the model.
- We will check the variable that has made the smallest or no change in the performance of the model, and then we will drop that variable or features; after that, we will be left with  $n-1$  features.
- Repeat the complete process until no feature can be dropped.

In this technique, by selecting the optimum performance of the model and maximum tolerable error rate, we can define the optimal number of features require for the machine learning algorithms.

## **Forward Feature Selection**

Forward feature selection follows the inverse process of the backward elimination process. It means, in this technique, we don't eliminate the feature; instead, we will find the best features that can produce the highest increase in the performance of the model. Below steps are performed in this technique:

- We start with a single feature only, and progressively we will add each feature at a time.
- Here we will train the model on each feature separately.
- The feature with the best performance is selected.
- The process will be repeated until we get a significant increase in the performance of the model.

## **Missing Value Ratio**

If a dataset has too many missing values, then we drop those variables as they do not carry much useful information. To perform this, we can set a threshold level, and if a variable has missing values more than that threshold, we will drop that variable. The higher the threshold value, the more efficient the reduction.

## **Low Variance Filter**

As same as missing value ratio technique, data columns with some changes in the data have less information. Therefore, we need to calculate the variance of each variable, and all data columns with variance lower than a given threshold are dropped because low variance features will not affect the target variable.

## High Correlation Filter

High Correlation refers to the case when two variables carry approximately similar information. Due to this factor, the performance of the model can be degraded. This correlation between the independent numerical variable gives the calculated value of the correlation coefficient. If this value is higher than the threshold value, we can remove one of the variables from the dataset. We can consider those variables or features that show a high correlation with the target variable.

## Random Forest

Random Forest is a popular and very useful feature selection algorithm in machine learning. This algorithm contains an in-built feature importance package, so we do not need to program it separately. In this technique, we need to generate a large set of trees against the target variable, and with the help of usage statistics of each attribute, we need to find the subset of features.

Random forest algorithm takes only numerical variables, so we need to convert the input data into numeric data using **hot encoding**.

## Factor Analysis

Factor analysis is a technique in which each variable is kept within a group according to the correlation with other variables, it means variables within a group can have a high correlation between themselves, but they have a low correlation with variables of other groups.

We can understand it by an example, such as if we have two variables Income and spend. These two variables have a high correlation, which means people with high income spends more, and vice versa. So, such variables are put into a group, and that group is known as the **factor**. The number of these factors will be reduced as compared to the original dimension of the dataset.

## Auto-encoders

One of the popular methods of dimensionality reduction is auto-encoder, which is a type of ANN or artificial neural network, and its main aim is to copy the inputs to their outputs. In this, the input is compressed into latent-space representation, and output is occurred using this representation. It has mainly two parts:

- **Encoder:** The function of the encoder is to compress the input to form the latent-space representation.
- **Decoder:** The function of the decoder is to recreate the output from the latent-space representation.

# OVERFITTING AND UNDERFITTING

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.

Before understanding the overfitting and underfitting, let's understand some basic term that will help to understand this topic well:

- **Signal:** It refers to the true underlying pattern of the data that helps the machine learning model to learn from the data.
- **Noise:** Noise is unnecessary and irrelevant data that reduces the performance of the model.
- **Bias:** Bias is a prediction error that is introduced in the model due to oversimplifying the machine learning algorithms. Or it is the difference between the predicted values and the actual values.
- **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

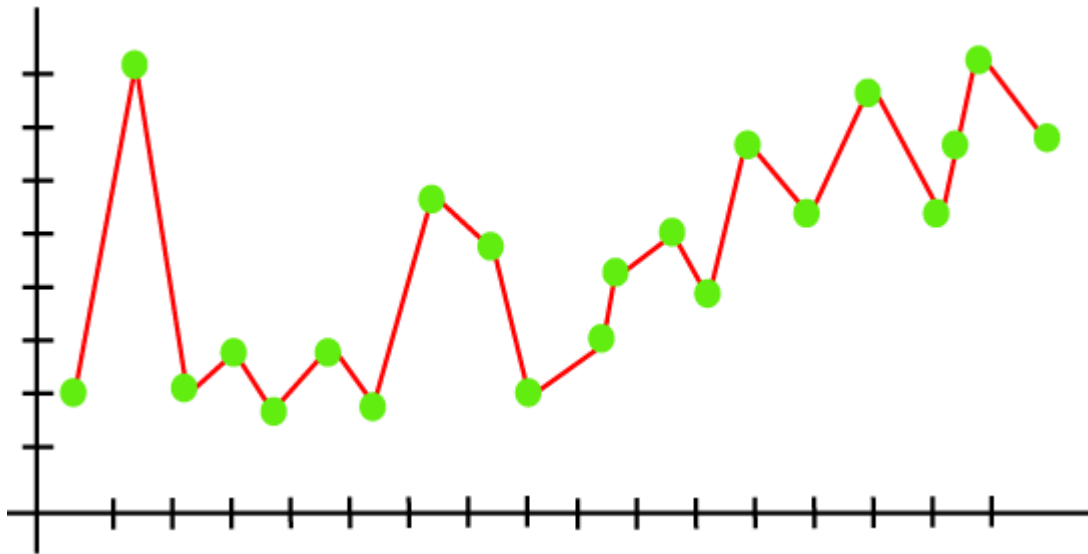
Overfitting occurs when our machine learning model tries to cover all the data points or more than the required data points present in the given dataset. Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy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

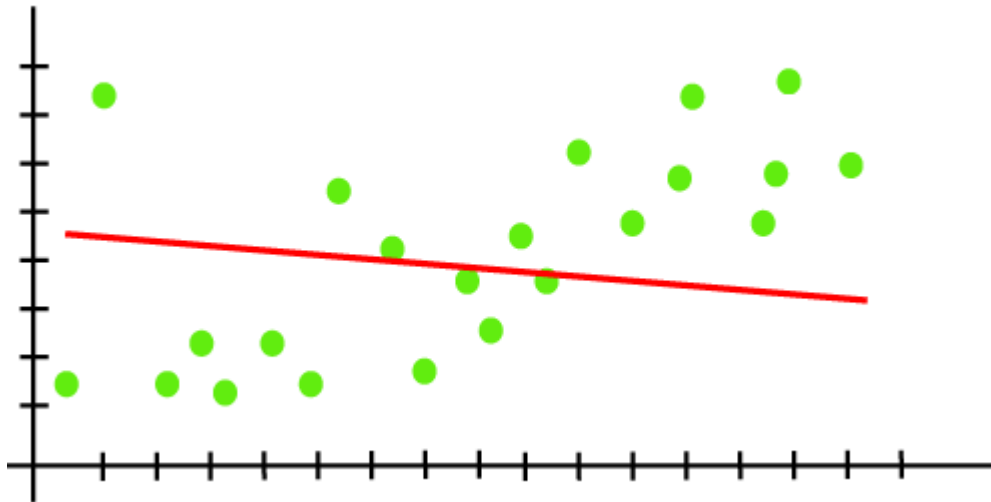
## Underfitting

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.

### Goodness of Fit

The "Goodness of fit" term is taken from the statistics, and the goal of the machine learning models to achieve the goodness of fit. In statistics modeling, *it defines how closely the result or predicted values match the true values of the dataset.*

The model with a good fit is between the underfitted and overfitted model, and ideally, it makes predictions with 0 errors, but in practice, it is difficult to achieve it.

As when we train our model for a time, the errors in the training data go down, and the same happens with test data. But if we train the model for a long duration, then the performance of the model may decrease due to the overfitting, as the model also learn the noise present in the dataset. The errors in the test dataset start increasing, so *the point, just before the raising of errors, is the good point, and we can stop here for achieving a good model.*

There are two other methods by which we can get a good point for our model, which are the **resampling method** to estimate model accuracy and **validation dataset**.

## Overfitting in Machine Learning

In the real world, the dataset present will never be clean and perfect. It means each dataset contains impurities, noisy data, outliers, missing data, or imbalanced data. Due to these impurities, different problems occur that affect the accuracy and the performance of the model. One of such problems is Overfitting in Machine Learning. *Overfitting is a problem that a model can exhibit.*

A statistical model is said to be overfitted if it can't generalize well with unseen data.

Before understanding overfitting, we need to know some basic terms, which are:

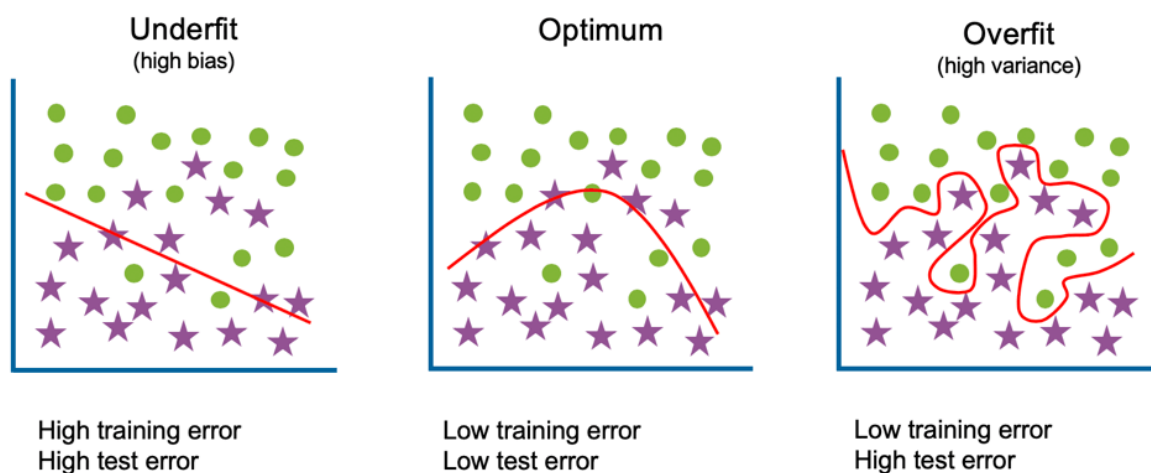
**Noise:** Noise is meaningless or irrelevant data present in the dataset. It affects the performance of the model if it is not removed.

**Bias:** Bias is a prediction error that is introduced in the model due to oversimplifying the machine learning algorithms. Or it is the difference between the predicted values and the actual values.

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

**Generalization:** It shows how well a model is trained to predict unseen data.

## What is Overfitting?



- Overfitting & underfitting are the two main errors/problems in the machine learning model, which cause poor performance in Machine Learning.

- Overfitting occurs when the model fits more data than required, and it tries to capture each and every datapoint fed to it. Hence it starts capturing noise and inaccurate data from the dataset, which degrades the performance of the model.
- An overfitted model doesn't perform accurately with the test/unseen dataset and can't generalize well.
- An overfitted model is said to have low bias and high variance.

## Example to Understand Overfitting

We can understand overfitting with a general example. Suppose there are three students, X, Y, and Z, and all three are preparing for an exam. X has studied only three sections of the book and left all other sections. Y has a good memory, hence memorized the whole book. And the third student, Z, has studied and practiced all the questions. So, in the exam, X will only be able to solve the questions if the exam has questions related to section 3. Student Y will only be able to solve questions if they appear exactly the same as given in the book. Student Z will be able to solve all the exam questions in a proper way.

The same happens with machine learning; if the algorithm learns from a small part of the data, it is unable to capture the required data points and hence under fitted.

Suppose the model learns the training dataset, like the Y student. They perform very well on the seen dataset but perform badly on unseen data or unknown instances. In such cases, the model is said to be Overfitting.

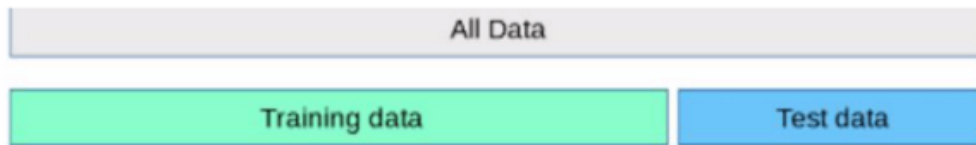
And if the model performs well with the training dataset and also with the test/unseen dataset, similar to student Z, it is said to be a good fit.

## How to detect Overfitting?

Overfitting in the model can only be detected once you test the data. To detect the issue, we can perform **Train/test split**.

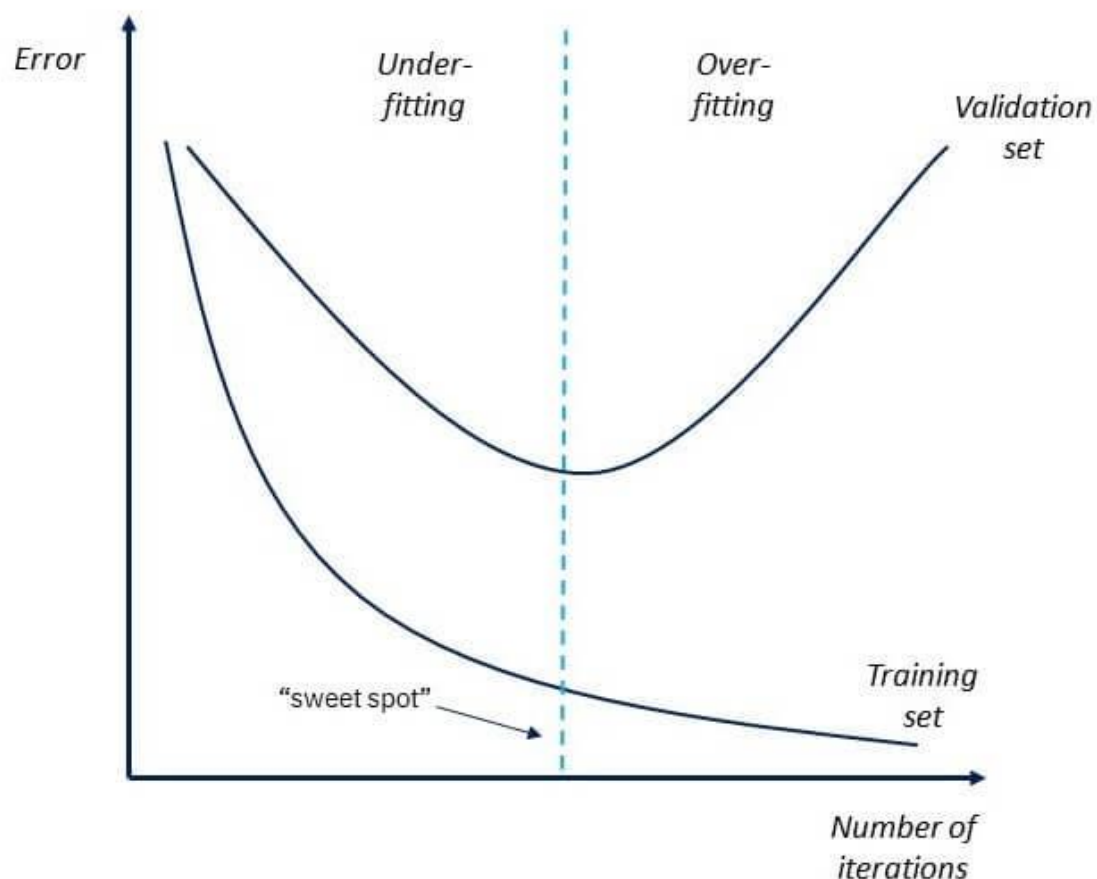
In the train-test split of the dataset, we can divide our dataset into random test and training datasets. We train the model with a training dataset which is about 80% of the total dataset. After training the model, we test it with the test dataset, which is 20 % of the total dataset.

## Train Test Split



Now, if the model performs well with the training dataset but not with the test dataset, then it is likely to have an overfitting issue.

For example, if the model shows 85% accuracy with training data and 50% accuracy with the test dataset, it means the model is not performing well.



## Ways to prevent the Overfitting

Although overfitting is an error in Machine learning which reduces the performance of the model, however, we can prevent it in several ways. With the use of the linear model, we can avoid overfitting; however, many real-world problems are non-linear

ones. It is important to prevent overfitting from the models. Below are several ways that can be used to prevent overfitting:

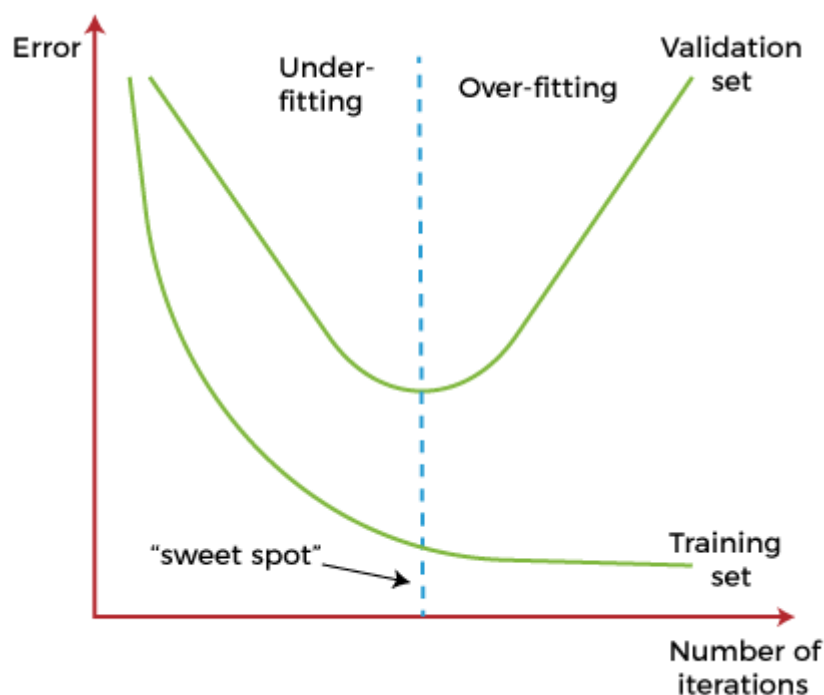
1. **Early Stopping**
2. **Train with more data**
3. **Feature Selection**
4. **Cross-Validation**
5. **Data Augmentation**
6. **Regularization**

## Early Stopping

In this technique, the training is paused before the model starts learning the noise within the model. In this process, while training the model iteratively, measure the performance of the model after each iteration. Continue up to a certain number of iterations until a new iteration improves the performance of the model.

After that point, the model begins to overfit the training data; hence we need to stop the process before the learner passes that point.

Stopping the training process before the model starts capturing noise from the data is known as **early stopping**.



However, this technique may lead to the underfitting problem if training is paused too early. So, it is very important to find that "sweet spot" between underfitting and overfitting.

## **Train with More data**

Increasing the training set by including more data can enhance the accuracy of the model, as it provides more chances to discover the relationship between input and output variables.

It may not always work to prevent overfitting, but this way helps the algorithm to detect the signal better to minimize the errors.

When a model is fed with more training data, it will be unable to overfit all the samples of data and forced to generalize well.

But in some cases, the additional data may add more noise to the model; hence we need to be sure that data is clean and free from in-consistencies before feeding it to the model.

## **Feature Selection**

While building the ML model, we have a number of parameters or features that are used to predict the outcome. However, sometimes some of these features are redundant or less important for the prediction, and for this feature selection process is applied. In the feature selection process, we identify the most important features within training data, and other features are removed. Further, this process helps to simplify the model and reduces noise from the data. Some algorithms have the auto-feature selection, and if not, then we can manually perform this process.

## **Cross-Validation**

Cross-validation is one of the powerful techniques to prevent overfitting.

In the general k-fold cross-validation technique, we divided the dataset into k-equal-sized subsets of data; these subsets are known as folds.

## **Data Augmentation**

Data Augmentation is a data analysis technique, which is an alternative to adding more data to prevent overfitting. In this technique, instead of adding more training data, slightly modified copies of already existing data are added to the dataset.

The data augmentation technique makes it possible to appear data sample slightly different every time it is processed by the model. Hence each data set appears unique to the model and prevents overfitting.

## **Regularization**

If overfitting occurs when a model is complex, we can reduce the number of features. However, overfitting may also occur with a simpler model, more specifically the Linear model, and for such cases, regularization techniques are much helpful.

Regularization is the most popular technique to prevent overfitting. It is a group of methods that forces the learning algorithms to make a model simpler. Applying the regularization technique may slightly increase the bias but slightly reduces the variance. In this technique, we modify the objective function by adding the penalizing term, which has a higher value with a more complex model.

The two commonly used regularization techniques are L1 Regularization and L2 Regularization.

## **Ensemble Methods**

In ensemble methods, prediction from different machine learning models is combined to identify the most popular result.

The most commonly used ensemble methods are **Bagging and Boosting**.

In bagging, individual data points can be selected more than once. After the collection of several sample datasets, these models are trained independently, and depending on the type of task-i.e., regression or classification-the average of those predictions is used to predict a more accurate result. Moreover, bagging reduces the chances of overfitting in complex models.

In boosting, a large number of weak learners arranged in a sequence are trained in such a way that each learner in the sequence learns from the mistakes of the learner before it. It combines all the weak learners to come out with one strong learner. In addition, it improves the predictive flexibility of simple models.



# PRINCIPAL COMPONENT ANALYSIS

Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. These new transformed features are called the **Principal Components**. It is one of the popular tools that is used for exploratory data analysis and predictive modeling. It is a technique to draw strong patterns from the given dataset by reducing the variances.

PCA generally tries to find the lower-dimensional surface to project the high-dimensional data.

PCA works by considering the variance of each attribute because the high attribute shows the good split between the classes, and hence it reduces the dimensionality. Some real-world applications of PCA are ***image processing, movie recommendation system, optimizing the power allocation in various communication channels***. It is a feature extraction technique, so it contains the important variables and drops the least important variable.

The PCA algorithm is based on some mathematical concepts such as:

- Variance and Covariance
- Eigenvalues and Eigen factors

## Some common terms used in PCA algorithm:

- **Dimensionality:** It is the number of features or variables present in the given dataset. More easily, it is the number of columns present in the dataset.
- **Correlation:** It signifies that how strongly two variables are related to each other. Such as if one changes, the other variable also gets changed. The correlation value ranges from -1 to +1. Here, -1 occurs if variables are inversely proportional to each other, and +1 indicates that variables are directly proportional to each other.
- **Orthogonal:** It defines that variables are not correlated to each other, and hence the correlation between the pair of variables is zero.
- **Eigenvectors:** If there is a square matrix  $M$ , and a non-zero vector  $v$  is given. Then  $v$  will be eigenvector if  $Av$  is the scalar multiple of  $v$ .
- **Covariance Matrix:** A matrix containing the covariance between the pair of variables is called the Covariance Matrix.

## Principal Components in PCA

As described above, the transformed new features or the output of PCA are the Principal Components. The number of these PCs are either equal to or less than the original features present in the dataset. Some properties of these principal components are given below:

- The principal component must be the linear combination of the original features.
- These components are orthogonal, i.e., the correlation between a pair of variables is zero.
- The importance of each component decreases when going to 1 to  $n$ , it means the 1 PC has the most importance, and  $n$  PC will have the least importance.

## Steps for PCA algorithm

### 1. Getting the dataset

Firstly, we need to take the input dataset and divide it into two subparts  $X$  and  $Y$ , where  $X$  is the training set, and  $Y$  is the validation set.

### 2. Representing data into a structure

Now we will represent our dataset into a structure. Such as we will represent the two-dimensional matrix of independent variable  $X$ . Here each row corresponds to the data items, and the column corresponds to the Features. The number of columns is the dimensions of the dataset.

### 3. Standardizing the data

In this step, we will standardize our dataset. Such as in a particular column, the features with high variance are more important compared to the features with lower variance. If the importance of features is independent of the variance of the feature, then we will divide each data item in a column with the standard deviation of the column. Here we will name the matrix as  $Z$ .

### 4. Calculating the Covariance of $Z$

To calculate the covariance of  $Z$ , we will take the matrix  $Z$ , and will transpose it. After transpose, we will multiply it by  $Z$ . The output matrix will be the Covariance matrix of  $Z$ .

### 5. Calculating the Eigen Values and Eigen Vectors

Now we need to calculate the eigenvalues and eigenvectors for the resultant covariance matrix  $Z$ . Eigenvectors or the covariance matrix are the directions of the axes with high information. And the coefficients of these eigenvectors are defined as the eigenvalues.

#### 6. **Sorting the Eigen Vectors**

In this step, we will take all the eigenvalues and will sort them in decreasing order, which means from largest to smallest. And simultaneously sort the eigenvectors accordingly in matrix P of eigenvalues. The resultant matrix will be named as  $P^*$ .

#### 7. **Calculating the new features Or Principal Components**

Here we will calculate the new features. To do this, we will multiply the  $P^*$  matrix to the Z. In the resultant matrix  $Z^*$ , each observation is the linear combination of original features. Each column of the  $Z^*$  matrix is independent of each other.

#### 8. **Remove less or unimportant features from the new dataset.**

The new feature set has occurred, so we will decide here what to keep and what to remove. It means, we will only keep the relevant or important features in the new dataset, and unimportant features will be removed out.

### **Applications of Principal Component Analysis**

- PCA is mainly used as the dimensionality reduction technique in various AI applications such **as computer vision, image compression, etc.**
- It can also be used for finding hidden patterns if data has high dimensions. Some fields where PCA is used are Finance, data mining, Psychology, etc.

# P-VALUE

In Statistical hypothesis testing, the P-value or sometimes called probability value, is used to observe the test results or more extreme results by assuming that the null hypothesis ( $H_0$ ) is true. In data science, there are lots of concepts that are borrowed from different disciplines, and the p-value is one of them. The concept of p-value comes from statistics and widely used in machine learning and data science.

- P-value is also used as an alternative to determine the point of rejection in order to provide the smallest significance level at which the null hypothesis is least or rejected.
- It is expressed as the level of significance that lies between 0 and 1, and *if there is smaller p-value, then there would be strong evidence to reject the null hypothesis*. If the value of p-value is very small, then it means the observed output is feasible but doesn't lie under the null hypothesis conditions ( $H_0$ ).
- The p-value of 0.05 is known as the level of significance ( $\alpha$ ). Usually, it is considered using two suggestions, which are given below:
  - **If  $p\text{-value} > 0.05$ :** The large p-value shows that the null hypothesis needs to be accepted.
  - **If  $p\text{-value} < 0.05$ :** The small p-value shows that the null hypothesis needs to be rejected, and the result is declared as statically significant.

In Statistics, our main goal is to determine the statistical significance of our result, and this statistical significance is made on below three concepts:

- Hypothesis Testing
- Normal Distribution
- Statistical Significance

## Hypothesis Testing

**Hypothesis testing** can be defined between two terms; **Null hypothesis** and **Alternative Hypothesis**. It is used to check the validity of the null hypothesis or claim made using the sample data. Here, the **null hypothesis ( $H_0$ )** is defined as the hypothesis with no statistical significance between two variables, while an **alternative hypothesis** is defined as the hypothesis with a statistical significance between the two variables. No significant relationship between the two variables tells that one variable will not affect the other variable. Thus, the Null hypothesis tells that what you are going to prove doesn't actually happen. If the independent variable doesn't affect the dependent variable, then it shows the alternative hypothesis condition.

In a simple way, we can say that *in hypothesis testing, first, we make a claim that is assumed as a null hypothesis using the sample data. If this claim is found invalid, then the alternative hypothesis is selected.* This assumption or claim is validated using the p-value to see if it is statistically significant or not using the evidence. If the evidence supports the alternative hypothesis, then the null hypothesis is rejected.

## Steps for Hypothesis testing

1. Claim or state a Null hypothesis for the experiment.
2. State the alternative hypothesis, which is opposite to the null hypothesis.
3. Set the value of alpha to be used in the experiment.
4. Determine the z-score using the normal distribution.
5. Compare the P-value to validate the statistical significance.

## Normal Distribution

The normal distribution, which is also known as Gaussian distribution, is the Probability distribution function. It is symmetric about the mean, and use to see the distribution of data using a graph plot. It shows that data near the mean is more frequent to occur as compared to data which is far from the mean, and it looks like a **bell-shaped curve**. The two main terms of the normal distribution are mean( $\mu$ ) and standard deviation( $\sigma$ ). For a normal distribution, the mean is zero, and the standard deviation is 1.

In hypothesis testing, we need to calculate z-score. **Z-score** is the number of standard deviations from the mean of data-point.

$$Z = \frac{x - \mu}{\sigma}$$

Here, the z-score inform us that where the data lies compared to the average population.

## Statistical significance:

To determine the statistical significance of the hypothesis test is the goal of calculating the p-value. To do this, first, we need to set a threshold, which is said to be alpha. We should always set the value of alpha before the experiment, and it is set to be either 0.05 or 0.01 (depending on the type of problem).

The result is concluded as a significant result if the observed p-value is lower than alpha.

## Errors in P-value

Two types of errors are defined for the p-value; these errors are given below:

1. Type I error
2. Type II error

### Type I Error:

It is defined as the incorrect or false rejection of the Null hypothesis. For this error, the maximum probability is alpha, and it is set in advance. The error is not affected by the sample size of the dataset. The type I error increases as we increase the number of tests or endpoints.

### Type II error

Type II error is defined as the wrong acceptance of the Null hypothesis. The probability of type II error is beta, and the beta depends upon the sample size and value of alpha. The beta cannot be determined as the function of the true population effect. The value of beta is inversely proportional to the sample size, and it means beta decreases as the sample size increases.

The value of beta also decreases when we increase the number of tests or endpoints.

	Decision	
Truth	Accept $H_0$	Reject $H_0$
$H_0$ is true	Correct decision	Type I error
$H_0$ is false	Type II error	Correct Decision

## Importance of P-value

The importance of p-value can be understood in two aspects:

- **Statistics Aspect:** In statistics, the concept of the p-value is important for hypothesis testing and statistical methods such as Regression.
- **Data Science Aspect:** In data science also, it is one of the important aspect Here the smaller p-value shows that there is an association between the predictor and response. It is advised while working with the machine learning problem in data science, the p-value should be taken carefully.

# REGULARIZATION

## What is Regularization?

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."*

## How does Regularization Work?

Regularization works by adding a penalty or complexity term to the complex model. Let's consider the simple linear regression equation:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \beta_n x_n + b$$

In the above equation, Y represents the value to be predicted

X<sub>1</sub>, X<sub>2</sub>, ...X<sub>n</sub> are the features for Y.

β<sub>0</sub>, β<sub>1</sub>, .....β<sub>n</sub> are the weights or magnitude attached to the features, respectively. Here represents the bias of the model, and b represents the intercept.

Linear regression models try to optimize the β<sub>0</sub> and b to minimize the cost function. The equation for the cost function for the linear model is given below:

$$\sum_{i=1}^M (y_i - y'_i)^2 = \sum_{i=1}^M (y_i - \sum_{j=0}^n \beta_j * X_{ij})^2$$

Now, we will add a loss function and optimize parameter to make the model that can predict the accurate value of Y. The loss function for the linear regression is called as **RSS or Residual sum of squares**.

## Techniques of Regularization

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

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

### Ridge Regression

- Ridge regression is one of the types of linear regression in which a small amount of bias is introduced so that we can get better long-term predictions.
- Ridge regression is a regularization technique, which is used to reduce the complexity of the model. It is also called as **L2 regularization**.
- In this technique, the cost function is altered by adding the penalty term to it. The amount of bias added to the model is called **Ridge Regression penalty**. We can calculate it by multiplying with the lambda to the squared weight of each individual feature.
- The equation for the cost function in ridge regression will be:

$$\sum_{i=1}^M (y_i - y'_i)^2 = \sum_{i=1}^M \left( y_i - \sum_{j=0}^n \beta_j * x_{ij} \right)^2 + \lambda \sum_{j=0}^n \beta_j^2$$

- In the above equation, the penalty term regularizes the coefficients of the model, and hence ridge regression reduces the amplitudes of the coefficients that decreases the complexity of the model.
- As we can see from the above equation, if the values of  **$\lambda$  tend to zero, the equation becomes the cost function of the linear regression model**. Hence, for the minimum value of  $\lambda$ , the model will resemble the linear regression model.
- A general linear or polynomial regression will fail if there is high collinearity between the independent variables, so to solve such problems, Ridge regression can be used.
- It helps to solve the problems if we have more parameters than samples.

### Lasso Regression:

- Lasso regression is another regularization technique to reduce the complexity of the model. It stands for **Least Absolute and Selection Operator**.
- It is similar to the Ridge Regression except that the penalty term contains only the absolute weights instead of a square of weights.



- Since it takes absolute values, hence, it can shrink the slope to 0, whereas Ridge Regression can only shrink it near to 0.
- It is also called as **L1 regularization**. The equation for the cost function of Lasso regression will be:

$$\sum_{i=1}^M (y_i - y'_i)^2 = \sum_{i=1}^M \left( y_i - \sum_{j=0}^n \beta_j * x_{ij} \right)^2 + \lambda \sum_{j=0}^n |\beta_j|$$

- Some of the features in this technique are completely neglected for model evaluation.
- Hence, the Lasso regression can help us to reduce the overfitting in the model as well as the feature selection.

### Key Difference between Ridge Regression and Lasso Regression

- **Ridge regression** is mostly used to reduce the overfitting in the model, and it includes all the features present in the model. It reduces the complexity of the model by shrinking the coefficients.
- **Lasso regression** helps to reduce the overfitting in the model as well as feature selection.

# ENCODING TECHNIQUES

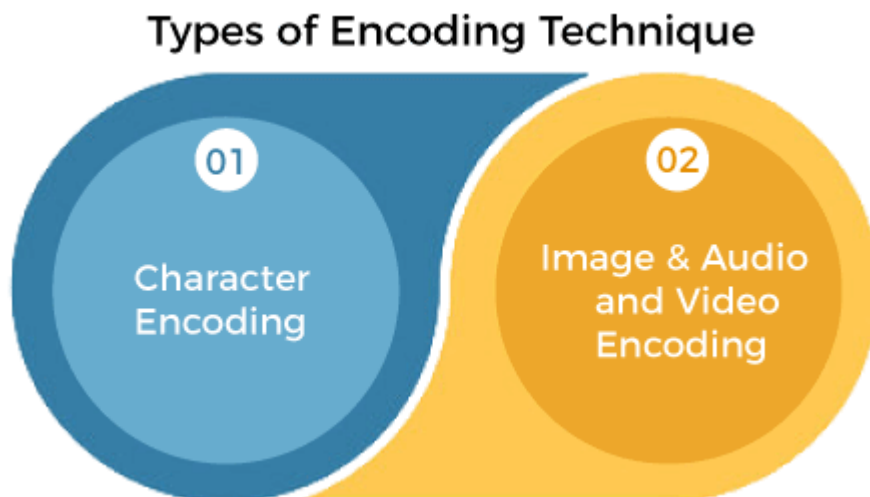
The process of conversion of data from one form to another form is known as *Encoding*. It is used to transform the data so that data can be supported and used by different systems. Encoding works similarly to converting temperature from centigrade to Fahrenheit, as it just gets converted in another form, but the original value always remains the same. Encoding is used in mainly two fields:

- **Encoding in Electronics:** In electronics, encoding refers to converting analog signals to digital signals.
- **Encoding in Computing:** In computing, encoding is a process of converting data to an equivalent cipher by applying specific code, letters, and numbers to the data.

Note: Encoding is different from encryption as its main purpose is not to hide the data but to convert it into a format so that it can be properly consumed.

In this topic, we are going to discuss the different types of encoding techniques that are used in computing.

## Type of Encoding Technique



- **Character Encoding**
- **Image & Audio and Video Encoding**

## Character Encoding

**Character encoding encodes characters into bytes.** It informs the computers how to interpret the zero and ones into real characters, numbers, and symbols. The computer understands only binary data; hence it is required to convert these characters into numeric codes. To achieve this, each character is converted into binary

code, and for this, text documents are saved with encoding types. It can be done by pairing numbers with characters. If we don't apply character encoding, our website will not display the characters and text in a proper format. Hence it will decrease the readability, and the machine would not be able to process data correctly. Further, character encoding makes sure that each character has a proper representation in computer or binary format.

There are different types of Character Encoding techniques, which are given below:

1. **HTML Encoding**
2. **URL Encoding**
3. **Unicode Encoding**
4. **Base64 Encoding**
5. **Hex Encoding**
6. **ASCII Encoding**

## **HTML Encoding**

HTML encoding is used to display an HTML page in a proper format. With encoding, a web browser gets to know that which character set to be used.

In HTML, there are various characters used in HTML Markup such as <, >. To encode these characters as content, we need to use an encoding.

## **URL Encoding**

URL (Uniform resource locator) Encoding is used to ***convert characters in such a format that they can be transmitted over the internet***. It is also known as percent-encoding. The URL Encoding is performed to send the URL to the internet using the ASCII character-set. Non-ASCII characters are replaced with a %, followed by the hexadecimal digits.

## **UNICODE Encoding**

Unicode is an encoding standard for a universal character set. It allows encoding, represent, and handle the text represented in most of the languages or writing systems that are available worldwide. It provides a code point or number for each character in every supported language. It can represent approximately all the possible characters possible in all the languages. A particular sequence of bits is known as a coding unit.

A UNICODE standard can use 8, 16, or 32 bits to represent the characters.

The Unicode standard defines Unicode Transformation Format (UTF) to encode the code points.

### **UNICODE Encoding standard has the following UTF schemes:**

- **UTF-8** **Encoding**  
The UTF8 is defined by the UNICODE standard, which is variable-width character encoding used in Electronics Communication. UTF-8 is capable of encoding all 1,112,064 valid character code points in Unicode using one to four one-byte (8-bit) code units.
- **UTF-16** **Encoding**  
UTF16 Encoding represents a character's code points using one of two 16-bits integers.
- **UTF-32** **Encoding**  
UTF32 Encoding represents each code point as 32-bit integers.

### **Base64 Encoding**

Base64 Encoding is used to encode binary data into equivalent ASCII Characters. The Base64 encoding is used in the Mail system as mail systems such as SMTP can't work with binary data because they accept ASCII textual data only. It is also used in simple HTTP authentication to encode the credentials. Moreover, it is also used to transfer the binary data into cookies and other parameters to make data unreadable to prevent tampering. If an image or another file is transferred without Base64 encoding, it will get corrupted as the mail system is not able to deal with binary data.

Base64 represents the data into blocks of 3 bytes, where each byte contains 8 bits; hence it represents 24 bits. These 24 bits are divided into four groups of 6 bits. Each of these groups or chunks are converted into equivalent Base64 value.

### **ASCII Encoding**

**American Standard Code for Information Interchange (ASCII)** is a type of character-encoding. It was the first character encoding standard released in the year 1963.

The ASCII code is used to represent English characters as numbers, where each letter is assigned with a number from **0 to 127**. Most modern character-encoding schemes are based on ASCII, though they support many additional characters. It is a single byte encoding only using the bottom 7 bits. In an ASCII file, each alphabetic, numeric, or special character is represented with a 7-bit binary number. Each character of the keyboard has an equivalent ASCII value.

## **Image and Audio & Video Encoding**

Image and audio & video encoding are performed to save storage space. A media file such as image, audio, and video are encoded to save them in a more efficient and compressed format.

These encoded files contain the same content with usually similar quality, but in compressed size, so that they can be saved within less space, can be transferred easily via mail, or can be downloaded on the system.

We can understand it as a . WAV audio file is converted into .MP3 file to reduce the size by  $1/10^{\text{th}}$  to its original size.

# FEATURE SELECTION TECHNIQUES

Feature selection is a way of selecting the subset of the most relevant features from the original features set by removing the redundant, irrelevant, or noisy features.

While developing the machine learning model, only a few variables in the dataset are useful for building the model, and the rest features are either redundant or irrelevant. If we input the dataset with all these redundant and irrelevant features, it may negatively impact and reduce the overall performance and accuracy of the model. Hence it is very important to identify and select the most appropriate features from the data and remove the irrelevant or less important features, which is done with the help of feature selection in machine learning.

Feature selection is one of the important concepts of machine learning, which highly impacts the performance of the model. As machine learning works on the concept of "Garbage In Garbage Out", so we always need to input the most appropriate and relevant dataset to the model in order to get a better result.

In this topic, we will discuss different feature selection techniques for machine learning. But before that, let's first understand some basics of feature selection.

- **What is Feature Selection?**
- **Need for Feature Selection**
- **Feature Selection Methods/Techniques**
- **Feature Selection statistics**

## What is Feature Selection?

A feature is an attribute that has an impact on a problem or is useful for the problem, and choosing the important features for the model is known as feature selection. Each machine learning process depends on feature engineering, which mainly contains two processes; which are Feature Selection and Feature Extraction. Although feature selection and extraction processes may have the same objective, both are completely different from each other. The main difference between them is that feature selection is about selecting the subset of the original feature set, whereas feature extraction creates new features. Feature selection is a way of reducing the input variable for the model by using only relevant data in order to reduce overfitting in the model.

So, we can define feature Selection as, "***It is a process of automatically or manually selecting the subset of most appropriate and relevant features to be used in model building.***" Feature selection is performed by either including the important features or excluding the irrelevant features in the dataset without changing them.

## Need for Feature Selection

Before implementing any technique, it is really important to understand, need for the technique and so for the Feature Selection. As we know, in machine learning, it is necessary to provide a pre-processed and good input dataset in order to get better outcomes. We collect a huge amount of data to train our model and help it to learn better. Generally, the dataset consists of noisy data, irrelevant data, and some part of useful data. Moreover, the huge amount of data also slows down the training process of the model, and with noise and irrelevant data, the model may not predict and perform well. So, it is very necessary to remove such noises and less-important data from the dataset and to do this, and Feature selection techniques are used.

Selecting the best features helps the model to perform well. For example, Suppose we want to create a model that automatically decides which car should be crushed for a spare part, and to do this, we have a dataset. This dataset contains a Model of the car, Year, Owner's name, Miles. So, in this dataset, the name of the owner does not contribute to the model performance as it does not decide if the car should be crushed or not, so we can remove this column and select the rest of the features(column) for the model building.

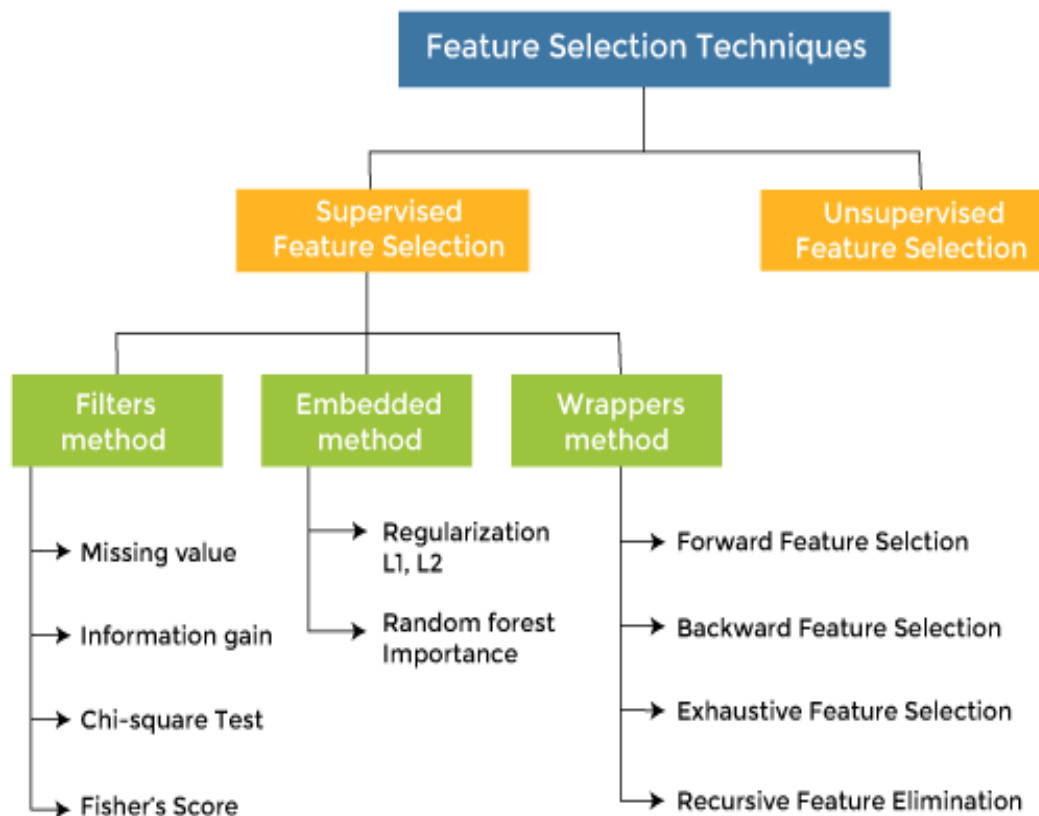
Below are some benefits of using feature selection in machine learning:

- **It helps in avoiding the curse of dimensionality.**
- **It helps in the simplification of the model so that it can be easily interpreted by the researchers.**
- **It reduces the training time.**
- **It reduces overfitting hence enhance the generalization.**

## Feature Selection Techniques

There are mainly two types of Feature Selection techniques, which are:

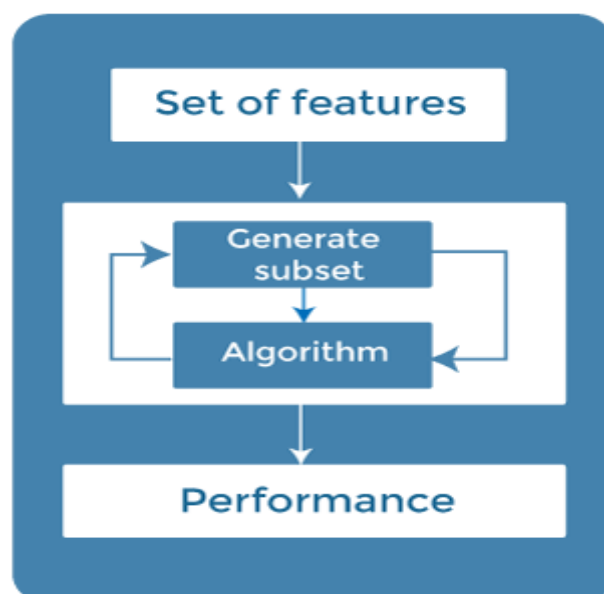
- **Supervised Feature Selection technique**  
Supervised Feature selection techniques consider the target variable and can be used for the labelled dataset.
- **Unsupervised Feature Selection technique**  
Unsupervised Feature selection techniques ignore the target variable and can be used for the unlabelled dataset.



There are mainly three techniques under supervised feature Selection:

## 1. Wrapper Methods

In wrapper methodology, selection of features is done by considering it as a search problem, in which different combinations are made, evaluated, and compared with other combinations. It trains the algorithm by using the subset of features iteratively.





On the basis of the output of the model, features are added or subtracted, and with this feature set, the model has trained again.

Some techniques of wrapper methods are:

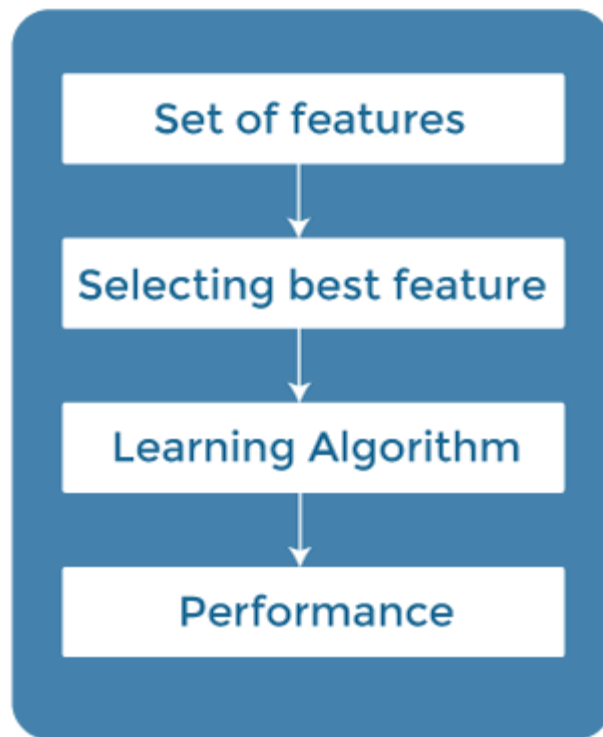
- **Forward selection** - Forward selection is an iterative process, which begins with an empty set of features. After each iteration, it keeps adding on a feature and evaluates the performance to check whether it is improving the performance or not. The process continues until the addition of a new variable/feature does not improve the performance of the model.
- **Backward elimination** - Backward elimination is also an iterative approach, but it is the opposite of forward selection. This technique begins the process by considering all the features and removes the least significant feature. This elimination process continues until removing the features does not improve the performance of the model.
- **Exhaustive Feature Selection**- Exhaustive feature selection is one of the best feature selection methods, which evaluates each feature set as brute-force. It means this method tries & make each possible combination of features and return the best performing feature set.
- **Recursive Feature Elimination**- Recursive feature elimination is a recursive greedy optimization approach, where features are selected by recursively taking a smaller and smaller subset of features. Now, an estimator is trained with each set of features, and the importance of each feature is determined using *coef\_attribute* or through a *feature\_importances\_attribute*.

## 2. Filter Methods

In Filter Method, features are selected on the basis of statistics measures. This method does not depend on the learning algorithm and chooses the features as a pre-processing step.

The filter method filters out the irrelevant feature and redundant columns from the model by using different metrics through ranking.

The advantage of using filter methods is that it needs low computational time and does not overfit the data.



Some common techniques of Filter methods are as follows:

- Information Gain
- Chi-square Test
- Fisher's Score
- Missing Value Ratio

**Information Gain:** Information gain determines the reduction in entropy while transforming the dataset. It can be used as a feature selection technique by calculating the information gain of each variable with respect to the target variable.

**Chi-square Test:** Chi-square test is a technique to determine the relationship between the categorical variables. The chi-square value is calculated between each feature and the target variable, and the desired number of features with the best chi-square value is selected.

**Fisher's Score:**

Fisher's score is one of the popular supervised technique of features selection. It returns the rank of the variable on the fisher's criteria in descending order. Then we can select the variables with a large fisher's score.

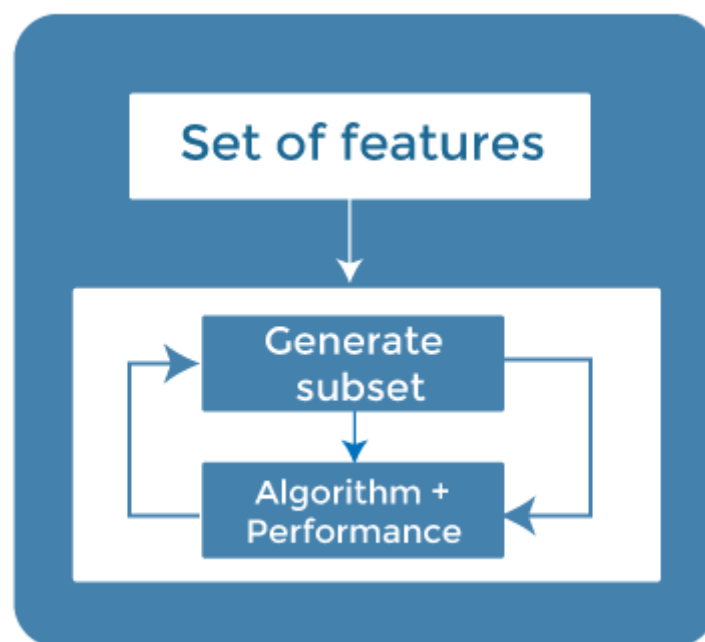
## Missing Value Ratio:

The value of the missing value ratio can be used for evaluating the feature set against the threshold value. The formula for obtaining the missing value ratio is the number of missing values in each column divided by the total number of observations. The variable is having more than the threshold value can be dropped.

$$\text{Missing Value Ratio} = \frac{\text{Number of Missing values} \times 100}{\text{Total number of observations}}$$

## 3. Embedded Methods

Embedded methods combined the advantages of both filter and wrapper methods by considering the interaction of features along with low computational cost. These are fast processing methods similar to the filter method but more accurate than the filter method.



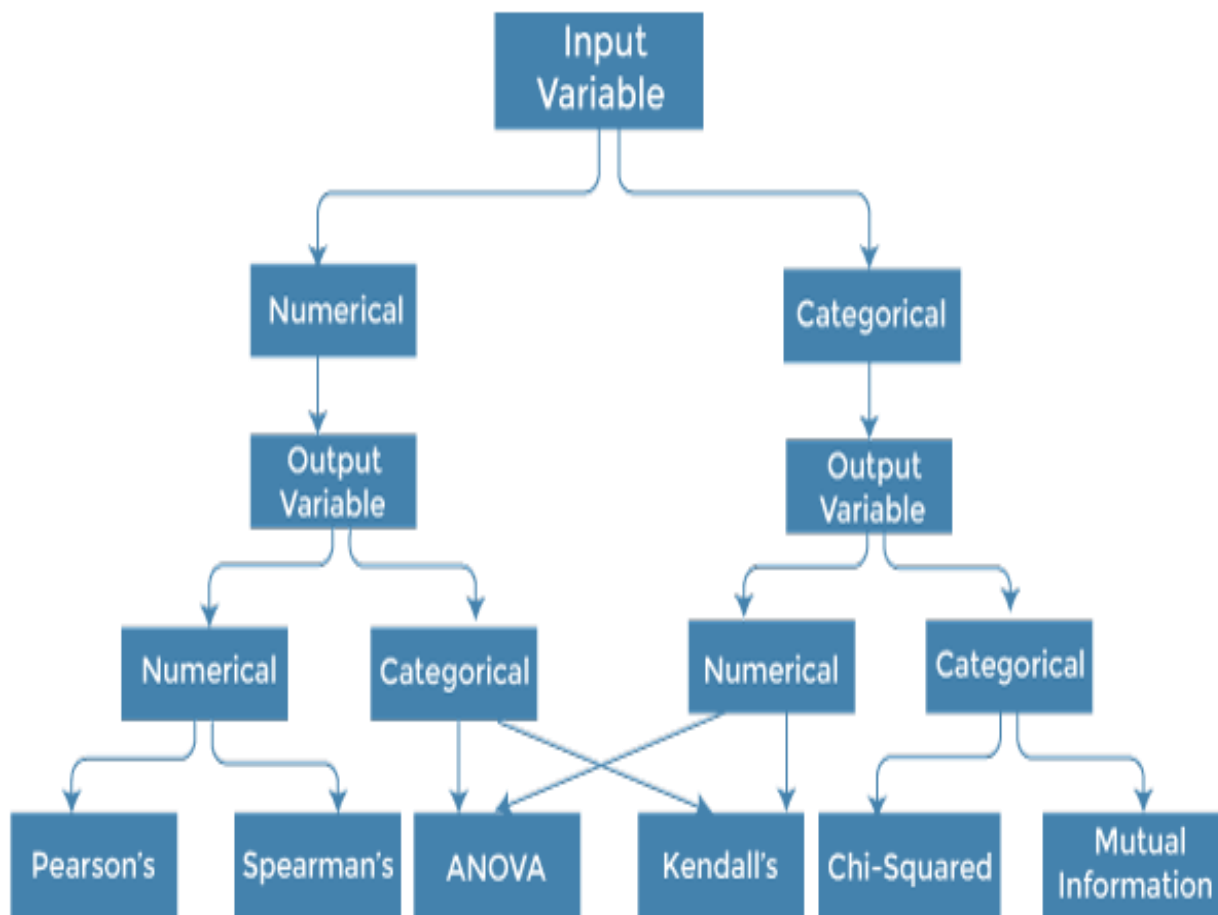
These methods are also iterative, which evaluates each iteration, and optimally finds the most important features that contribute the most to training in a particular iteration. Some techniques of embedded methods are:

- **Regularization-** Regularization adds a penalty term to different parameters of the machine learning model for avoiding overfitting in the model. This penalty term is added to the coefficients; hence it shrinks some coefficients to zero. Those features with zero coefficients can be removed from the dataset. The types of regularization techniques are L1 Regularization (Lasso Regularization) or Elastic Nets (L1 and L2 regularization).

- **Random Forest Importance** - Different tree-based methods of feature selection help us with feature importance to provide a way of selecting features. Here, feature importance specifies which feature has more importance in model building or has a great impact on the target variable. Random Forest is such a tree-based method, which is a type of bagging algorithm that aggregates a different number of decision trees. It automatically ranks the nodes by their performance or decrease in the impurity (Gini impurity) over all the trees. Nodes are arranged as per the impurity values, and thus it allows to pruning of trees below a specific node. The remaining nodes create a subset of the most important features.

## How to choose a Feature Selection Method?

For machine learning engineers, it is very important to understand that which feature selection method will work properly for their model. The more we know the datatypes of variables, the easier it is to choose the appropriate statistical measure for feature selection.



To know this, we need to first identify the type of input and output variables. In machine learning, variables are of mainly two types:

- **Numerical Variables:** Variable with continuous values such as integer, float
- **Categorical Variables:** Variables with categorical values such as Boolean, ordinal, nominals.

Below are some univariate statistical measures, which can be used for filter-based feature selection:

### **1. Numerical Input, Numerical Output:**

Numerical Input variables are used for predictive regression modelling. The common method to be used for such a case is the Correlation coefficient.

- Pearson's correlation coefficient (For linear Correlation).
- Spearman's rank coefficient (for non-linear correlation).

### **2. Numerical Input, Categorical Output:**

Numerical Input with categorical output is the case for classification predictive modelling problems. In this case, also, correlation-based techniques should be used, but with categorical output.

- **ANOVA correlation coefficient (linear).**
- **Kendall's rank coefficient (nonlinear).**

### **3. Categorical Input, Numerical Output:**

This is the case of regression predictive modelling with categorical input. It is a different example of a regression problem. We can use the same measures as discussed in the above case but in reverse order.

### **4. Categorical Input, Categorical Output:**

This is a case of classification predictive modelling with categorical Input variables.

The commonly used technique for such a case is Chi-Squared Test. We can also use Information gain in this case.

We can summarise the above cases with appropriate measures in the below table:

Input Variable	Output Variable	Feature Selection technique
Numerical	Numerical	<ul style="list-style-type: none"><li>○ Pearson's correlation coefficient (For linear Correlation).</li><li>○ Spearman's rank coefficient (for non-linear correlation).</li></ul>
Numerical	Categorical	<ul style="list-style-type: none"><li>○ ANOVA correlation coefficient (linear).</li><li>○ Kendall's rank coefficient (nonlinear).</li></ul>
Categorical	Numerical	<ul style="list-style-type: none"><li>○ Kendall's rank coefficient (linear).</li><li>○ ANOVA correlation coefficient (nonlinear).</li></ul>
Categorical	Categorical	<ul style="list-style-type: none"><li>○ Chi-Squared test (contingency tables).</li><li>○ Mutual Information.</li></ul>

## Conclusion

Feature selection is a very complicated and vast field of machine learning, and lots of studies are already made to discover the best methods. There is no fixed rule of the best feature selection method. However, choosing the method depend on a machine learning engineer who can combine and innovate approaches to find the best method for a specific problem. One should try a variety of model fits on different subsets of features selected through different statistical Measures.

# BIAS AND VARIANCE

Machine learning is a branch of Artificial Intelligence, which allows machines to perform data analysis and make predictions. However, if the machine learning model is not accurate, it can make prediction errors, and these prediction errors are usually known as Bias and Variance. In machine learning, these errors will always be present as there is always a slight difference between the model predictions and actual predictions. The main aim of ML/data science analysts is to reduce these errors in order to get more accurate results. In this topic, we are going to discuss bias and variance, Bias-variance trade-off, Underfitting and Overfitting. But before starting, let's first understand what errors in Machine learning are?

## Errors in Machine Learning?

In machine learning, an error is a measure of how accurately an algorithm can make predictions for the previously unknown dataset. On the basis of these errors, the machine learning model is selected that can perform best on the particular dataset. There are mainly two types of errors in machine learning, which are:

- **Reducible errors:** These errors can be reduced to improve the model accuracy. Such errors can further be classified into bias and Variance.
- **Irreducible errors:** These errors will always be present in the model

regardless of which algorithm has been used. The cause of these errors is unknown variables whose value can't be reduced.

## What is 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.*** It can be defined as an inability of machine learning algorithms such as Linear Regression to capture the true relationship between the data points. Each algorithm begins with some amount of bias because bias occurs from assumptions in the model, which makes the target function simple to learn. A model has either:

- **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. **A high bias model also cannot perform well on new data.**

Generally, a linear algorithm has a high bias, as it makes them learn fast. The simpler the algorithm, the higher the bias it has likely to be introduced. Whereas a nonlinear algorithm often has low bias.

Some examples of machine learning algorithms with low bias are **Decision Trees, k-Nearest Neighbours and Support Vector Machines**. At the same time, an algorithm with high bias is **Linear Regression, Linear Discriminant Analysis and Logistic Regression**.

### Ways to reduce High Bias:

High bias mainly occurs due to a much simple model. Below are some ways to reduce the high bias:

- Increase the input features as the model is underfitted.
- Decrease the regularization term.
- Use more complex models, such as including some polynomial features.

### What is a Variance Error?

The variance would specify the amount of variation in the prediction if the different training data was used. In simple words, ***variance tells that how much a random variable is different from its expected value.*** Ideally, a model should not vary too much from one training dataset to another, which means the algorithm should be good in understanding the hidden mapping between inputs and output variables. Variance errors are either of **low variance or high variance**.

**Low variance** means there is a small variation in the prediction of the target function with changes in the training data set. At the same time, **High variance** shows a large variation in the prediction of the target function with changes in the training dataset.

A model that shows high variance learns a lot and perform well with the training dataset, and does not generalize well with the unseen dataset. As a result, such a model gives good results with the training dataset but shows high error rates on the test dataset.

Since, with high variance, the model learns too much from the dataset, it leads to overfitting of the model. A model with high variance has the below problems:

- A high variance model leads to overfitting.
- Increase model complexities.

Usually, nonlinear algorithms have a lot of flexibility to fit the model, have high variance.



Some examples of machine learning algorithms with low variance are, **Linear Regression, Logistic Regression, and Linear discriminant analysis**. At the same time, algorithms with high variance are **decision tree, Support Vector Machine, and K-nearest neighbours**.

### Ways to Reduce High Variance:

- Reduce the input features or number of parameters as a model is overfitted.
- Do not use a much complex model.
- Increase the training data.
- Increase the Regularization term.

### Different Combinations of Bias-Variance

There are four possible combinations of bias and variances, which are represented by the below diagram:

1. **Low-Bias, Low-Variance:**

The combination of low bias and low variance shows an ideal machine learning model. However, it is not possible practically.

2. **Low-Bias, High-Variance:**

With low bias and high variance, model predictions are inconsistent and accurate on average. This case occurs when the model learns with a large number of parameters and hence leads to an **overfitting**

3. **High-Bias, Low-Variance:**

With High bias and low variance, predictions are consistent but inaccurate on average. This case occurs when a model does not learn well with the training dataset or uses few numbers of the parameter. It leads to **underfitting** problems in the model.

4. **High-Bias, High-Variance:**

With high bias and high variance, predictions are inconsistent and also inaccurate on average.

### How to identify High variance or High Bias?

High variance can be identified if the model has:

- Low training error and high test error.

High Bias can be identified if the model has:

- High training error and the test error is almost similar to training error.

## **Bias-Variance Trade-Off**

While building the machine learning model, it is really important to take care of bias and variance in order to avoid overfitting and underfitting in the model. If the model is very simple with fewer parameters, it may have low variance and high bias. Whereas, if the model has a large number of parameters, it will have high variance and low bias. So, it is required to make a balance between bias and variance errors, and this balance between the bias error and variance error is known as **the Bias-Variance trade-off**.

For an accurate prediction of the model, algorithms need a low variance and low bias. But this is not possible because bias and variance are related to each other:

- If we decrease the variance, it will increase the bias.
- If we decrease the bias, it will increase the variance.

Bias-Variance trade-off is a central issue in supervised learning. Ideally, we need a model that accurately captures the regularities in training data and simultaneously generalizes well with the unseen dataset. Unfortunately, doing this is not possible simultaneously. Because a high variance algorithm may perform well with training data, but it may lead to overfitting to noisy data. Whereas, high bias algorithm generates a much simple model that may not even capture important regularities in the data. So, we need to find a sweet spot between bias and variance to make an optimal model.

Hence, the ***Bias-Variance trade-off is about finding the sweet spot to make a balance between bias and variance errors.***

# MACHINE LEARNING TOOLS

Machine learning is one of the most revolutionary technologies that is making lives simpler. ***It is a subfield of Artificial Intelligence, which analyses the data, build the model, and make predictions.*** Due to its popularity and great applications, every tech enthusiast wants to learn and build new machine learning Apps. However, to build ML models, it is important to master machine learning tools. Mastering machine learning tools will enable you to play with the data, train your models, discover new methods, and create algorithms.

There are different tools, software, and platform available for machine learning, and also new software and tools are evolving day by day. Although there are many options and availability of Machine learning tools, choosing the best tool per your model is a challenging task. If you choose the right tool for your model, you can make it faster and more efficient. In this topic, we will discuss some popular and commonly used Machine learning tools and their features.



## 1. TensorFlow

TensorFlow is one of the most popular open-source libraries used to train and build both machine learning and deep learning models. It provides a JS library and was developed by **Google Brain Team**. It is much popular among machine learning enthusiasts, and they use it for building different ML applications. It offers a powerful library, tools, and resources for numerical computation, specifically for large scale machine learning and deep learning projects. It enables data scientists/ML developers to build and deploy machine learning applications efficiently. For training and building the ML models, TensorFlow provides a high-level Keras API, which lets users easily start with TensorFlow and machine learning.



### Features:

Below are some top features:

- TensorFlow enables us to build and train our ML models easily.
- It also enables you to run the existing models using **the TensorFlow.js**
- It provides multiple abstraction levels that allow the user to select the correct resource as per the requirement.
- It helps in building a neural network.
- Provides support of distributed computing.
- While building a model, for more need of flexibility, it provides eager execution that enables immediate iteration and intuitive debugging.
- This is open-source software and highly flexible.
- It also enables the developers to perform numerical computations using data flow graphs.
- Run-on GPUs and CPUs, and also on various mobile computing platforms.
- It provides a functionality of auto diff (Automatically computing gradients is called automatic differentiation or auto diff).
- It enables to easily deploy and training the model in the cloud.
- It can be used in two ways, i.e., by installing through NPM or by script tags.
- It is free to use.

## 2. PyTorch

PyTorch is an open-source machine learning framework, which is based on **the Torch** library. This framework is free and open-source and developed by **FAIR(Facebook's AI Research lab)**. It is one of the popular ML frameworks, which

can be used for various applications, including computer vision and natural language processing. PyTorch has Python and C++ interfaces; however, the Python interface is more interactive. Different deep learning software is made up on top of PyTorch, such as PyTorch Lightning, Hugging Face's Transformers, Tesla autopilot, etc.



It specifies a Tensor class containing an n-dimensional array that can perform tensor computations along with GPU support.

### **Features:**

Below are some top features:

- It enables the developers to create neural networks using Autograd Module.
- It is more suitable for deep learning researches with good speed and flexibility.
- It can also be used on cloud platforms.
- It includes tutorial courses, various tools, and libraries.
- It also provides a dynamic computational graph that makes this library more popular.
- It allows changing the network behaviour randomly without any lag.
- It is easy to use due to its hybrid front-end.
- It is freely available.

### **3. Google Cloud ML Engine**

While training a classifier with a huge amount of data, a computer system might not perform well. However, various machine learning or deep learning projects require millions or billions of training datasets. Or the algorithm that is being used is taking a long time for execution. In such a case, one should go for the Google Cloud ML Engine. It is a hosted platform where ML developers and data scientists build and run optimum quality machine learning models. It provides a managed service that allows developers to easily create ML models with any type of data and of any size.



### **Features:**

Below are the top features:

- Provides machine learning model training, building, deep learning and predictive modelling.

- The two services, namely, prediction and training, can be used independently or combinedly.
- It can be used by enterprises, i.e., for identifying clouds in a satellite image, responding faster to emails of customers.
- It can be widely used to train a complex model.

#### 4. Amazon Machine Learning (AML)

Amazon provides a great number of machine learning tools, and one of them is **Amazon Machine Learning** or AML. Amazon Machine Learning (AML) is a cloud-based and robust machine learning software application, which is widely used for building machine learning models and making predictions. Moreover, it integrates data from multiple sources, including **Redshift, Amazon S3, or RDS**.



#### Features

Below are some top features:

- AML offers visualization tools and wizards.
- Enables the users to identify the patterns, build mathematical models, and make predictions.
- It provides support for three types of models, which are multi-class classification, binary classification, and regression.
- It permits users to import the model into or export the model out from Amazon Machine Learning.
- It also provides core concepts of machine learning, including ML models, Data sources, Evaluations, Real-time predictions and Batch predictions.
- It enables the user to retrieve predictions with the help of batch APIs for bulk requests or real-time APIs for individual requests.

#### 5. NET

Accord.Net is .Net based Machine Learning framework, which is used for scientific computing. It is combined with audio and image processing libraries that are written in C#. This framework provides different libraries for various applications in ML, such as **Pattern Recognition, linear algebra, Statistical Data processing**. One popular package of the Accord.Net framework is **Accord. Statistics, Accord.Math, and Accord.MachineLearning**.



## Features

Below are some top features:

- It contains 38+ kernel Functions.
- Consists of more than 40 non-parametric and parametric estimation of statistical distributions.
- Used for creating production-grade computer audition, computer vision, signal processing, and statistics apps.
- Contains more than 35 hypothesis tests that include two-way and one way ANOVA tests, non-parametric tests such as the Kolmogorov-Smirnov test and many more.

## 6. Apache Mahout

Apache Mahout is an open-source project of Apache Software Foundation, which is used for developing machine learning applications mainly focused on Linear Algebra. It is a distributed linear algebra framework and mathematically expressive Scala DSL, which enable the developers to promptly implement their own algorithms. It also provides Java/Scala libraries to perform Mathematical operations mainly based on linear algebra and statistics.

### Features:

Below are some top features:

- It enables developers to implement machine learning techniques, including recommendation, clustering, and classification.
- It is an efficient framework for implementing scalable algorithms.
- It consists of matrix and vector libraries.
- It provides support for multiple distributed backends(including Apache Spark)
- It runs on top of Apache Hadoop using the MapReduce paradigm.

## 7. Shogun

Shogun is a free and open-source machine learning software library, which was created by **Gunnar Raetsch and Soeren Sonnenburg** in the year **1999**. This software library is written in C++ and supports interfaces for different languages such as Python, R, Scala, C#, Ruby, etc., using **SWIG**(Simplified Wrapper and Interface Generator). The main aim of Shogun is on different kernel-based algorithms such as Support Vector Machine (SVM), K-Means Clustering, etc., for



regression and classification problems. It also provides the complete implementation of Hidden Markov Models.

## Features:

Below are some top features:

- The main aim of Shogun is on different kernel-based algorithms such as Support Vector Machine (SVM), K-Means Clustering, etc., for regression and classification problems.
- It provides support for the use of pre-calculated kernels.
- It also offers to use a combined kernel using Multiple kernel Learning Functionality.
- This was initially designed for processing a huge dataset that consists of up to 10 million samples.
- It also enables users to work on interfaces on different programming languages such as Lua, Python, Java, C#, Octave, Ruby, MATLAB, and R.

## 8. Oryx2

It is a realization of the lambda architecture and built on **Apache Kafka** and **Apache Spark**. It is widely used for real-time large-scale machine learning projects. It is a framework for building apps, including end-to-end applications for filtering, packaged, regression, classification, and clustering. It is written in Java languages, including Apache Spark, Hadoop, Tomcat, Kafka, etc. The latest version of Oryx2 is Oryx 2.8.0.



## Features:

Below are some top features:

- It has three tiers: specialization on top providing ML abstractions, generic lambda architecture tier, end-to-end implementation of the same standard ML algorithms.
- The original project of Oryx2 was Oryx1, and after some upgrades, Oryx2 was launched.
- It is well suited for large-scale real-time machine learning projects.
- It contains three layers which are arranged side-by-side, and these are named as Speed layer, batch layer, and serving layer.
- It also has a data transport layer that transfer data between different layers and receives input from external sources.



## 9. Apache Spark MLlib

Apache Spark MLlib is a scalable machine learning library that runs on Apache Mesos, Hadoop, Kubernetes, standalone, or in the cloud. Moreover, it can access data from different data sources. It is an open-source cluster-computing framework that offers an interface for complete clusters along with data parallelism and fault tolerance.



For optimized numerical processing of data, MLlib provides linear algebra packages such as Breeze and netlib-Java. It uses a query optimizer and physical execution engine for achieving high performance with both batch and streaming data.

### Features

Below are some top features:

- MLlib contains various algorithms, including Classification, Regression, Clustering, recommendations, association rules, etc.
- It runs different platforms such as Hadoop, Apache Mesos, Kubernetes, standalone, or in the cloud against diverse data sources.
- It contains high-quality algorithms that provide great results and performance.
- It is easy to use as it provides interfaces In Java, Python, Scala, R, and SQL.

## 10. Google ML kit for Mobile

For Mobile app developers, Google brings ML Kit, which is packaged with the expertise of machine learning and technology to create more robust, optimized, and personalized apps. This tools kit can be used for face detection, text recognition, landmark detection, image labelling, and barcode scanning applications. One can also use it for working offline.



### Features:

Below are some top features:

- The ML kit is optimized for mobile.
- It includes the advantages of different machine learning technologies.
- It provides easy-to-use APIs that enables powerful use cases in your mobile apps.

- It includes Vision API and Natural Language APIS to detect faces, text, and objects, and identify different languages & provide reply suggestions.

## **Conclusion**

In this topic, we have discussed some popular machine learning tools. However, there are many more other ML tools, but choosing the tool completely depends on the requirement for one's project, skills, and price to the tool. Most of these tools are freely available, except for some tools such as Rapid Miner. Each tool works in a different language and provides some specifications.

# GRADIENT DESCENT

Gradient Descent is known as one of the most commonly used optimization algorithms to train machine learning models by means of minimizing errors between actual and expected results. Further, gradient descent is also used to train Neural Networks.

In mathematical terminology, Optimization algorithm refers to the task of minimizing/maximizing an objective function  $f(x)$  parameterized by  $x$ . Similarly, in machine learning, optimization is the task of minimizing the cost function parameterized by the model's parameters. The main objective of gradient descent is to minimize the convex function using iteration of parameter updates. Once these machine learning models are optimized, these models can be used as powerful tools for Artificial Intelligence and various computer science applications.

In this tutorial on Gradient Descent in Machine Learning, we will learn in detail about gradient descent, the role of cost functions specifically as a barometer within Machine Learning, types of gradient descents, learning rates, etc.

## What is Gradient Descent or Steepest Descent?

Gradient descent was initially discovered by "**Augustin-Louis Cauchy**" in mid of 18th century. ***Gradient Descent is defined as one of the most commonly used iterative optimization algorithms of machine learning to train the machine learning and deep learning models. It helps in finding the local minimum of a function.***

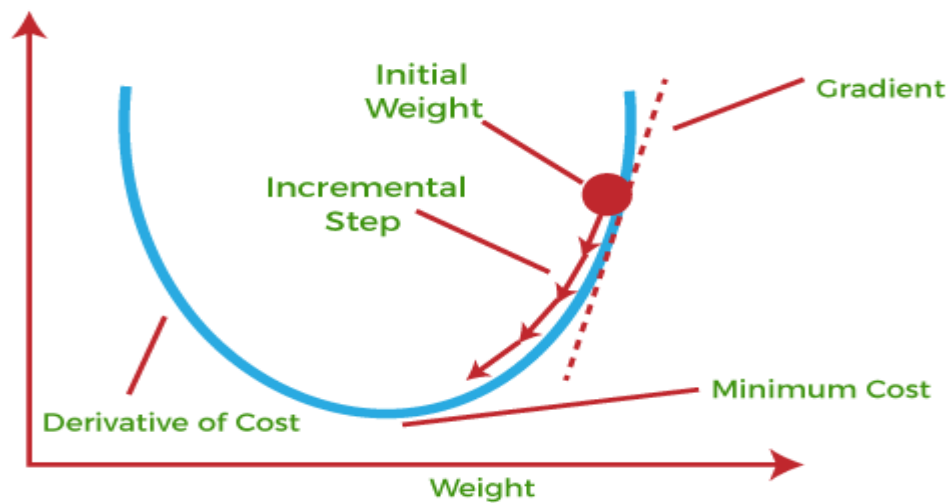
The best way to define the local minimum or local maximum of a function using gradient descent is as follows:

- If we move towards a negative gradient or away from the gradient of the function at the current point, it will give the **local minimum** of that function.
- Whenever we move towards a positive gradient or towards the gradient of the function at the current point, we will get the **local maximum** of that function.

This entire procedure is known as Gradient Ascent, which is also known as steepest descent. ***The main objective of using a gradient descent algorithm is to minimize the cost function using iteration.*** To achieve this goal, it performs two steps iteratively:

- Calculates the first-order derivative of the function to compute the gradient or slope of that function.
- Move away from the direction of the gradient, which means slope increased from the current point by alpha times, where Alpha is defined as Learning Rate.

It is a tuning parameter in the optimization process which helps to decide the length of the steps.



## What is Cost-function?

***The cost function is defined as the measurement of difference or error between actual values and expected values at the current position and present in the form of a single real number.*** It helps to increase and improve machine learning efficiency by providing feedback to this model so that it can minimize error and find the local or global minimum. Further, it continuously iterates along the direction of the negative gradient until the cost function approaches zero. At this steepest descent point, the model will stop learning further. Although cost function and loss function are considered synonymous, also there is a minor difference between them. The slight difference between the loss function and the cost function is about the error within the training of machine learning models, as loss function refers to the error of one training example, while a cost function calculates the average error across an entire training set.

The cost function is calculated after making a hypothesis with initial parameters and modifying these parameters using gradient descent algorithms over known data to reduce the cost function.

Hypothesis:

Parameters:

Cost function:

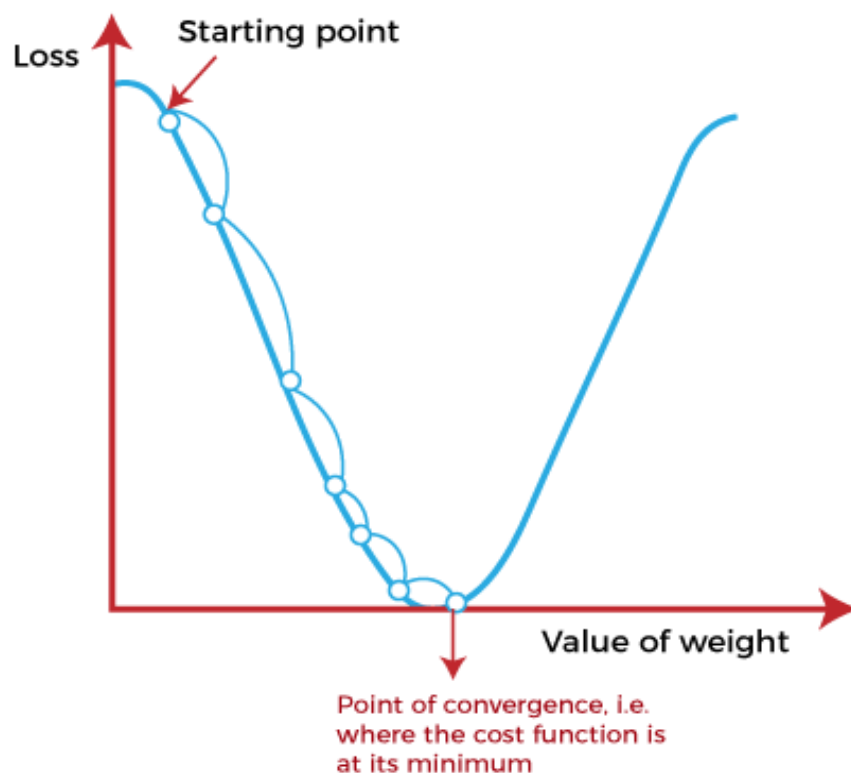
Goal:

## How does Gradient Descent work?

Before starting the working principle of gradient descent, we should know some basic concepts to find out the slope of a line from linear regression. The equation for simple linear regression is given as:

$$Y=mX+c$$

Where 'm' represents the slope of the line, and 'c' represents the intercepts on the y-axis.



The starting point(shown in above fig.) is used to evaluate the performance as it is considered just as an arbitrary point. At this starting point, we will derive the first derivative or slope and then use a tangent line to calculate the steepness of this slope. Further, this slope will inform the updates to the parameters (weights and bias).

The slope becomes steeper at the starting point or arbitrary point, but whenever new parameters are generated, then steepness gradually reduces, and at the lowest point, it approaches the lowest point, which is called **a point of convergence**.

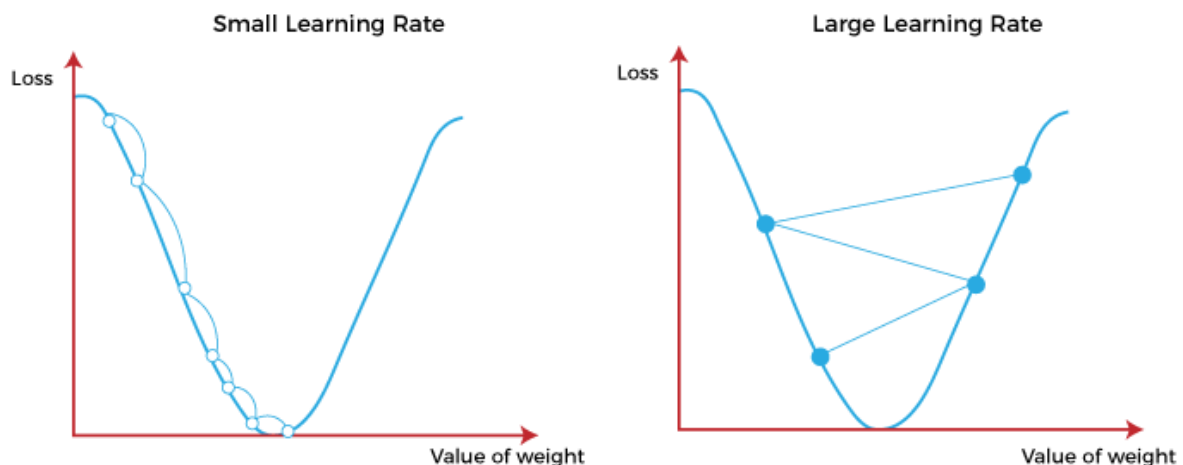
The main objective of gradient descent is to minimize the cost function or the error between expected and actual. To minimize the cost function, two data points are required:

- **Direction & Learning Rate**

These two factors are used to determine the partial derivative calculation of future iteration and allow it to the point of convergence or local minimum or global minimum. Let's discuss learning rate factors in brief;

### **Learning Rate:**

It is defined as the step size taken to reach the minimum or lowest point. This is typically a small value that is evaluated and updated based on the behavior of the cost function. If the learning rate is high, it results in larger steps but also leads to risks of overshooting the minimum. At the same time, a low learning rate shows the small step sizes, which compromises overall efficiency but gives the advantage of more precision.



### **Types of Gradient Descent**

Based on the error in various training models, the Gradient Descent learning algorithm can be divided into **Batch gradient descent, stochastic gradient descent, and mini-batch gradient descent**. Let's understand these different types of gradient descent:

#### **1. Batch Gradient Descent:**

Batch gradient descent (BGD) is used to find the error for each point in the training set and update the model after evaluating all training examples. This procedure is known as the training epoch. In simple words, it is a greedy approach where we have to sum over all examples for each update.

#### **Advantages of Batch gradient descent:**

- It produces less noise in comparison to other gradient descent.

- It produces stable gradient descent convergence.
- It is Computationally efficient as all resources are used for all training samples.

## **2. Stochastic gradient descent**

Stochastic gradient descent (SGD) is a type of gradient descent that runs one training example per iteration. Or in other words, it processes a training epoch for each example within a dataset and updates each training example's parameters one at a time. As it requires only one training example at a time, hence it is easier to store in allocated memory. However, it shows some computational efficiency losses in comparison to batch gradient systems as it shows frequent updates that require more detail and speed. Further, due to frequent updates, it is also treated as a noisy gradient. However, sometimes it can be helpful in finding the global minimum and also escaping the local minimum.

### **Advantages of Stochastic gradient descent:**

In Stochastic gradient descent (SGD), learning happens on every example, and it consists of a few advantages over other gradient descent.

- It is easier to allocate in desired memory.
- It is relatively fast to compute than batch gradient descent.
- It is more efficient for large datasets.

## **3. Mini Batch Gradient Descent:**

Mini Batch gradient descent is the combination of both batch gradient descent and stochastic gradient descent. It divides the training datasets into small batch sizes then performs the updates on those batches separately. Splitting training datasets into smaller batches make a balance to maintain the computational efficiency of batch gradient descent and speed of stochastic gradient descent. Hence, we can achieve a special type of gradient descent with higher computational efficiency and less noisy gradient descent.

### **Advantages of Mini Batch gradient descent:**

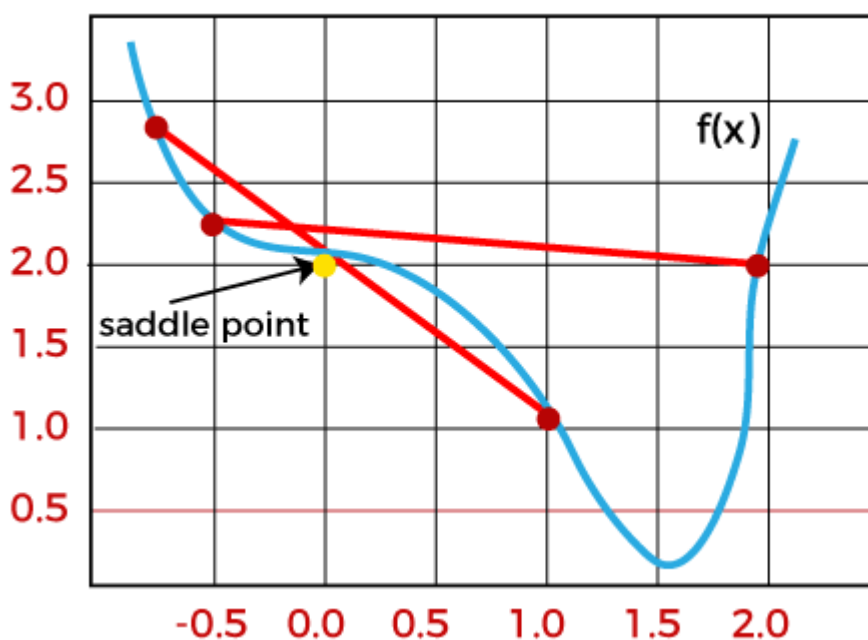
- It is easier to fit in allocated memory.
- It is computationally efficient.
- It produces stable gradient descent convergence.

## Challenges with the Gradient Descent

Although we know Gradient Descent is one of the most popular methods for optimization problems, it still also has some challenges. There are a few challenges as follows:

### 1. Local Minima and Saddle Point:

For convex problems, gradient descent can find the global minimum easily, while for non-convex problems, it is sometimes difficult to find the global minimum, where the machine learning models achieve the best results.



Whenever the slope of the cost function is at zero or just close to zero, this model stops learning further. Apart from the global minimum, there occur some scenarios that can show this slop, which is saddle point and local minimum. Local minima generate the shape similar to the global minimum, where the slope of the cost function increases on both sides of the current points.

In contrast, with saddle points, the negative gradient only occurs on one side of the point, which reaches a local maximum on one side and a local minimum on the other side. The name of a saddle point is taken by that of a horse's saddle.

The name of local minima is because the value of the loss function is minimum at that point in a local region. In contrast, the name of the global minima is given so because the value of the loss function is minimum there, globally across the entire domain the loss function.



## **2. Vanishing and Exploding Gradient**

In a deep neural network, if the model is trained with gradient descent and backpropagation, there can occur two more issues other than local minima and saddle point.

### **Vanishing Gradients:**

Vanishing Gradient occurs when the gradient is smaller than expected. During backpropagation, this gradient becomes smaller that causing the decrease in the learning rate of earlier layers than the later layer of the network. Once this happens, the weight parameters update until they become insignificant.

### **Exploding Gradient:**

Exploding gradient is just opposite to the vanishing gradient as it occurs when the Gradient is too large and creates a stable model. Further, in this scenario, model weight increases, and they will be represented as NaN. This problem can be solved using the dimensionality reduction technique, which helps to minimize complexity within the model.

# TIME SERIES ANALYSIS

Time Series Analysis is a way of studying the characteristics of the response variable concerning time as the independent variable. To estimate the target variable in predicting or forecasting, use the time variable as the reference point. TSA represents a series of time-based orders, it would be Years, Months, Weeks, Days, Hours, Minutes, and Seconds. It is an observation from the sequence of discrete time of successive intervals. Some real-world application of TSA includes weather forecasting models, stock market predictions, signal processing, and control systems. Since TSA involves producing the set of information in a particular sequence, this makes it distinct from spatial and other analyses. We could predict the future using AR, MA, ARMA, and ARIMA models. In this article, we will be decoding time series analysis for you.

## What Is Time Series Analysis?

Time series analysis is a specific way of analyzing a sequence of data points collected over time. In TSA, analysts record data points at consistent intervals over a set period rather than just recording the data points intermittently or randomly.

## Objectives of Time Series Analysis

- To understand how time series works and what factors affect a certain variable(s) at different points in time.
- Time series analysis will provide the consequences and insights of the given dataset's features that change over time.
- Supporting to derive the predicting the future values of the time series variable.
- Assumptions: There is only one assumption in TSA, which is "stationary," which means that the origin of time does not affect the properties of the process under the statistical factor.

## How to Analyze Time Series?

To perform the time series analysis, we have to follow the following steps:

- Collecting the data and cleaning it
- Preparing Visualization with respect to time vs key feature
- Observing the stationarity of the series
- Developing charts to understand its nature.
- Model building – AR, MA, ARMA and ARIMA
- Extracting insights from prediction

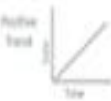





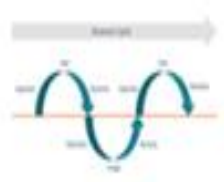
## Significance of Time Series

TSA is the backbone for prediction and forecasting analysis, specific to time-based problem statements.

- Analyzing the historical dataset and its patterns
- Understanding and matching the current situation with patterns derived from the previous stage.
- Understanding the factor or factors influencing certain variable(s) in different periods.

With the help of “Time Series,” we can prepare numerous time-based analyses and results.

- Forecasting: Predicting any value for the future.
- Segmentation: Grouping similar items together.
- Classification: Classifying a set of items into given classes.
- Descriptive analysis: Analysis of a given dataset to find out what is there in it.
- Intervention analysis: Effect of changing a given variable on the outcome.

	Trend	Seasonality	Cyclical	Irregularity
Time	Fixed Time Interval	Fixed Time Interval	Not Fixed Time Interval	Not Fixed Time Interval
Duration	Long and Short Term	Short Term	Long and Short Term	Regular/Irregular
Visualization				
Nature - I	Gradual	Swings between Up or Down	Repeating Up and Down	Errored or High Fluctuation
Nature – II	Upward/Down Trend	Pattern repeatable	No fixed period	Short and Not repeatable
Prediction Capability	Predictable	Predictable	Challenging	Challenging
Market Model				Highly random/Unforeseen Events – along with white noise.

## Components of Time Series Analysis

Let's look at the various components of Time Series Analysis:

- **Trend:** In which there is no fixed interval and any divergence within the given dataset is a continuous timeline. The trend would be Negative or Positive or Null Trend
- **Seasonality:** In which regular or fixed interval shifts within the dataset in a continuous timeline. Would be bell curve or saw tooth
- **Cyclical:** In which there is no fixed interval, uncertainty in movement and its pattern
- **Irregularity:** Unexpected situations/events/scenarios and spikes in a short time span.

## What Are the Limitations of Time Series Analysis?

Time series has the below-mentioned limitations; we have to take care of those during our data analysis.

- Similar to other models, the missing values are not supported by TSA
- The data points must be linear in their relationship.
- Data transformations are mandatory, so they are a little expensive.
- Models mostly work on Uni-variate data.

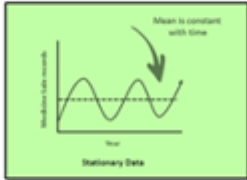
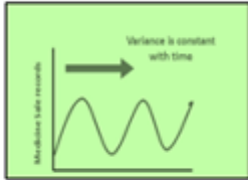

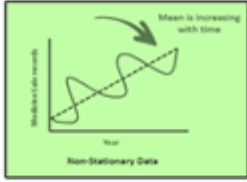

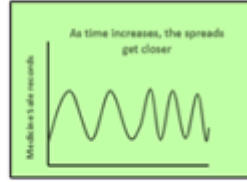
## Data Types of Time Series

Let's discuss the time series' data types and their influence. While discussing TS data types, there are two major types – stationary and non-stationary.

**Stationary:** A dataset should follow the below thumb rules without having Trend, Seasonality, Cyclical, and Irregularity components of the time series.

- The mean value of them should be completely constant in the data during the analysis.
- The variance should be constant with respect to the time-frame
- Covariance measures the relationship between two variables.

**Non- Stationary:** If either the mean-variance or covariance is changing with respect to time, the dataset is called non-stationary.

	MEAN	Variance	Covariance
Stationary	 <p>Mean is constant with time</p>	 <p>Variance is constant with time</p>	 <p>Mean value and difference between two spreads is time-independent</p>
Non-Stationary	 <p>Mean is increasing with time</p>	 <p>Variance is varying with time</p>	 <p>As time increases, the spreads get closer</p>

Designed by Author (Shanthababu)

## Methods to Check Stationarity

During the TSA model preparation workflow, we must assess whether the dataset is stationary or not. This is done using Statistical Tests. There are two tests available to test if the dataset is stationary:

- Augmented Dickey-Fuller (ADF) Test
- Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test

### Augmented Dickey-Fuller (ADF) Test or Unit Root Test:

The ADF test is the most popular statistical test. It is done with the following assumptions:

- Null Hypothesis (H0): Series is non-stationary
- Alternate Hypothesis (HA): Series is stationary
  - p-value > 0.05 Fail to reject (H0)
  - p-value ≤ 0.05 Accept (H1)

### Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test:

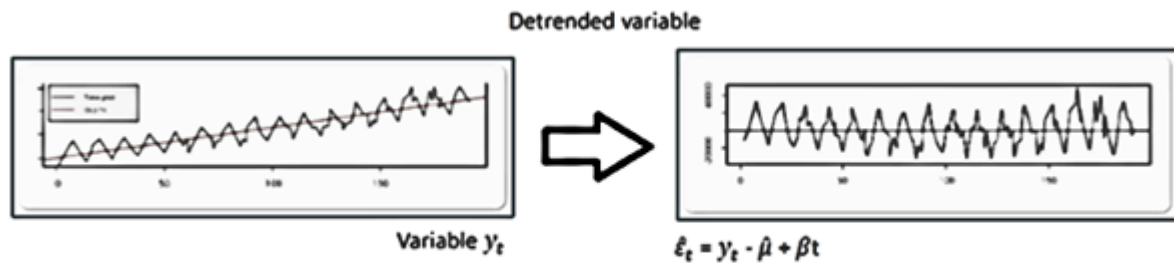
These tests are used for testing a NULL Hypothesis (H0) that will perceive the time series as stationary around a deterministic trend against the alternative of a unit root. Since TSA is looking for Stationary Data for its further analysis, we have to ensure that the dataset is stationary.

## Converting Non-Stationary Into Stationary

Let's discuss quickly how to convert non-stationary to stationary for effective time series modeling. There are three methods available for this conversion – detrending, differencing, and transformation.

## Detrending

It involves removing the trend effects from the given dataset and showing only the differences in values from the trend. It always allows cyclical patterns to be identified.

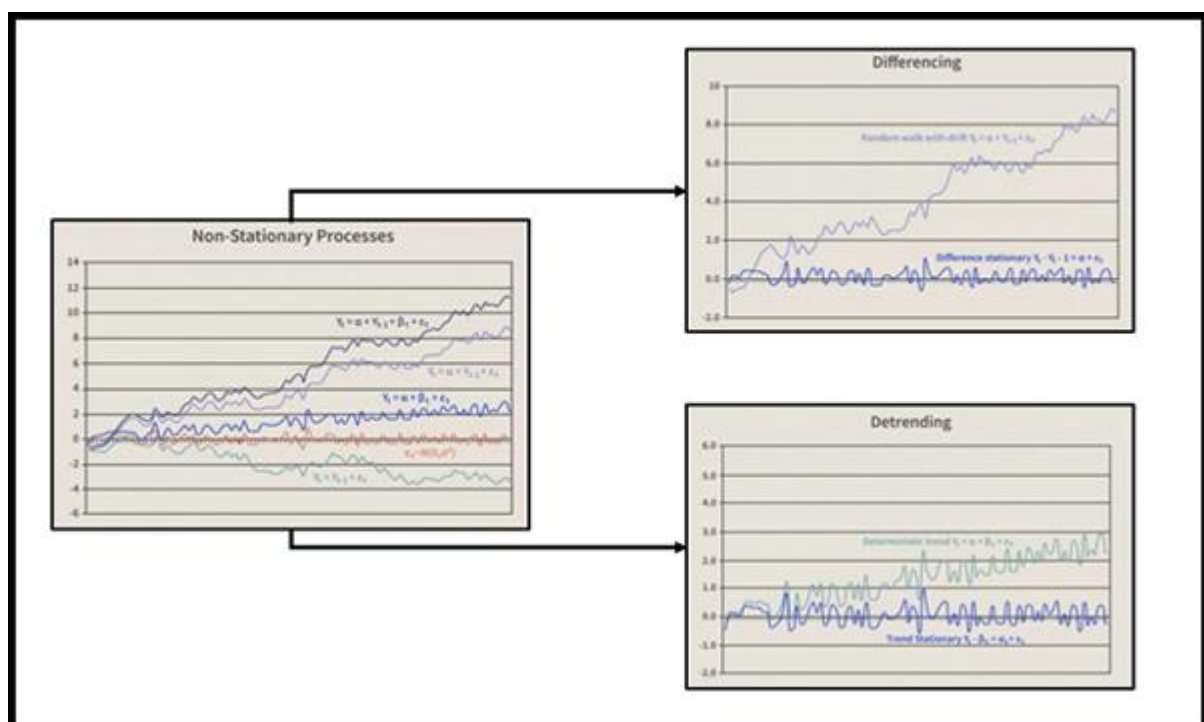


Designed by Author (Shanthababu)

## Differencing

This is a simple transformation of the series into a new time series, which we use to remove the series dependence on time and stabilize the mean of the time series, so trend and seasonality are reduced during this transformation.

- $Y_t = Y_t - Y_{t-1}$
- $Y_t = \text{Value with time}$



## Transformation

This includes three different methods they are Power Transform, Square Root, and Log Transfer. The most commonly used one is Log Transfer.

## Moving Average Methodology

The commonly used time series method is the Moving Average. This method is slick with random short-term variations. Relatively associated with the components of time series.

The Moving Average (MA) (or) Rolling Mean: The value of MA is calculated by taking average data of the time-series within k periods.

Let's see the types of moving averages:

- Simple Moving Average (SMA),
- Cumulative Moving Average (CMA)
- Exponential Moving Average (EMA)

### Simple Moving Average (SMA)

The Simple Moving Average (SMA) calculates the unweighted mean of the previous M or N points. We prefer selecting sliding window data points based on the amount of smoothing, as increasing the value of M or N improves smoothing but reduces accuracy.

To understand better, I will use the air temperature dataset.

$$SMA_t = \frac{x_t + x_{t-1} + x_{t-2} + \dots + x_{M-(t-1)}}{M}$$

	A	N	O	P	Q	R
1	Any	Avg Temp SMA				
2	1780	14.075				
3	1781	14.71667				
4	1782	13.63333	=(N2+N3+N4)/3			
5	1783	14.4	14.25			
6	1784	13.61667	13.88333			
7	1785	14.15833	14.05833			
8	1786	14.19167				
9	1787	14.025				
10	1788	14.275				
11	1789	13.91667				
12	1790	14.45833				
13	1791	14.44167				
14	1792	14.64167				
15	1793	14.29167				
16	1794	14.66667				
17	1795	14.25833				

## Cumulative Moving Average (CMA):

The CMA is the unweighted mean of past values till the current time.

$$CMA_t = \frac{x_1 + x_2 + x_3 + \dots + x_t}{t}$$

	A	N	O	P	Q
1	Any	Avg Temp SMA		CMV	
2	1780	14.075		14.075	
3	1781	14.71667		14.39583	
4	1782	13.63333	14.14167	14.14167	
5	1783	14.4	14.25	$=(N2+N3+N4+N5)/4$	
6	1784	13.61667	13.88333	14.08833	
7	1785	14.15833	14.05833	14.1	
8	1786	14.19167			
9	1787	14.025			
10	1788	14.275			
11	1789	13.91667			
12	1790	14.45833			
13	1791	14.44167			
14	1792	14.64167			
15	1793	14.29167			
16	1794	14.66667			
17	1795	14.25833			
18	1796	13.75			
19	1797	14.04167			
20	1798	15.075			
21	1799	14.5			
22	1800	14.18333			
23	1801	14			

temperature\_TSA

## Exponential Moving Average (EMA):

EMA is mainly used to identify trends and filter out noise. The weight of elements is decreased gradually over time. This means It gives weight to recent data points, not historical ones. Compared with SMA, the EMA is faster to change and more sensitive.

$\alpha$  → Smoothing Factor.

- It has a value between 0,1.
- Represents the weighting applied to the very recent period.

Let's apply the exponential moving averages with a smoothing factor of 0.1 and 0.3 in the given dataset.



$$EMA_t = \begin{cases} x_0 & t = 0 \\ \alpha x_t + (1 - \alpha) EMA_{t-1} & t > 0 \end{cases}$$

	A	N	O	P	Q	R	S
1	Any	Avg Temp	SMA	CMV	EMA		
2	1780	14.075		14.075	14.075		
3	1781	14.71667		14.39583	14.13917		
4	1782	13.63333	14.14167	14.14167	14.60833		
5	1783	14.4	14.25	14.20625	13.71		
6	1784	13.61667	13.88333	14.08833	14.32167		
7	1785	14.15833	14.05833	14.1	=0.1*N7+0.9*N6		
8	1786	14.19167					
9	1787	14.025					
10	1788	14.275					
11	1789	13.91667					
12	1790	14.45833					
13	1791	14.44167					
14	1792	14.64167					
15	1793	14.29167					
16	1794	14.66667					
17	1795	14.25833					
18	1796	13.75					
19	1797	14.04167					
20	1798	15.075					
21	1799	14.5					
22	1800	14.18333					
23	1801	14					

temperature\_TSA

## Time Series Analysis in Data Science and Machine Learning

When dealing with TSA in Data Science and Machine Learning, there are multiple model options are available. In which the Autoregressive–Moving-Average (ARMA) models with [p, d, and q].

- P==> autoregressive lags
- q== moving average lags
- d==> difference in the order

Before we get to know about Arima, first, you should understand the below terms better.

- Auto-Correlation Function (ACF)
- Partial Auto-Correlation Function (PACF)

### Auto-Correlation Function (ACF)

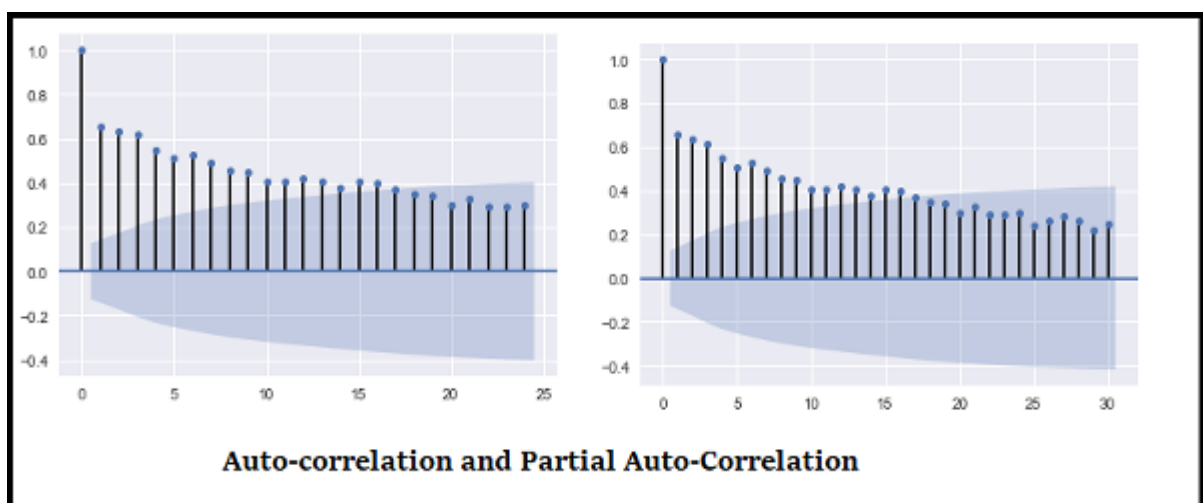
ACF indicates how similar a value is within a given time series and the previous value. (OR) It measures the degree of the similarity between a given time series and the lagged version of that time series at the various intervals we observed.

Python Statsmodels library calculates autocorrelation. It identifies a set of trends in the given dataset and the influence of former observed values on the currently observed values.

## Partial Auto-Correlation (PACF)

PACF is similar to Auto-Correlation Function and is a little challenging to understand. It always shows the correlation of the sequence with itself with some number of time units per sequence order in which only the direct effect has been shown, and all other intermediary effects are removed from the given time series.

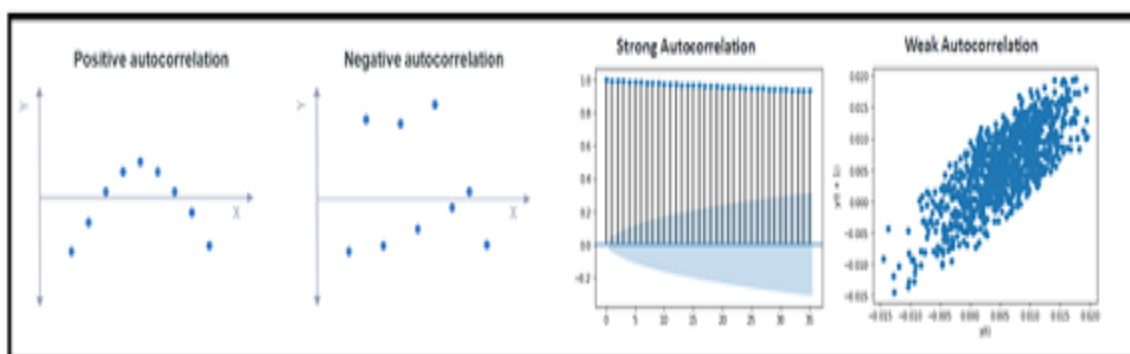
## Auto-Correlation and Partial Auto-Correlation



## Observation

The previous temperature influences the current temperature, but the significance of that influence decreases and slightly increases from the above visualization along with the temperature with regular time intervals.

## Types of Auto-Correlation



## Interpret ACF and PACF Plots

ACF	PACF	Perfect ML -Model
Plot declines gradually	Plot drops instantly	Auto Regressive model.
Plot drops instantly	Plot declines gradually	Moving Average model
Plot decline gradually	Plot Decline gradually	ARMA
Plot drop instantly	Plot drop instantly	You wouldn't perform any model

Remember that both ACF and PACF require stationary time series for analysis.

## What Is an Auto-Regressive Model?

An auto-regressive model is a simple model that predicts future performance based on past performance. It is mainly used for forecasting when there is some correlation between values in a given time series and those that precede and succeed (back and forth).

An AR is a Linear Regression model that uses lagged variables as input. By indicating the input, the Linear Regression model can be easily built using the scikit-learn library. Statsmodels library provides autoregression model-specific functions where you must specify an appropriate lag value and train the model. It is provided in the AutoTeg class to get the results using simple steps.

- Creating the model AutoReg()
- Call fit() to train it on our dataset.
- Returns an AutoRegResults object.
- Once fit, make a prediction by calling the predict () function

The equation for the AR model (Let's compare  $Y=mX+c$ )

$$Y_t = C + b_1 Y_{t-1} + b_2 Y_{t-2} + \dots + b_p Y_{t-p} + E_t$$

## Key Parameters

- $p$ =past values
- $Y_t$ =Function of different past values
- $E_t$ =errors in time
- $C$ =intercept

## Understanding ARMA and ARIMA

ARMA is a combination of the Auto-Regressive and Moving Average models for forecasting. This model provides a weakly stationary stochastic process in terms of two polynomials, one for the Auto-Regressive and the second for the Moving Average.

$$Y_t = \underbrace{\mu + \sum_{i=1}^q \gamma_i Y_{t-i}}_{\text{Auto-Regressive}} + \underbrace{\varepsilon_t + \sum_{i=1}^q \theta_i \varepsilon_{t-i}}_{\text{Moving Average}}$$

ARMA is best for predicting stationary series. ARIMA was thus developed to support both stationary as well as non-stationary series.



- AR ==> Uses past values to predict the future.
- MA ==> Uses past error terms in the given series to predict the future.
- I==> Uses the differencing of observation and makes the stationary data.

AR+I+MA= ARIMA

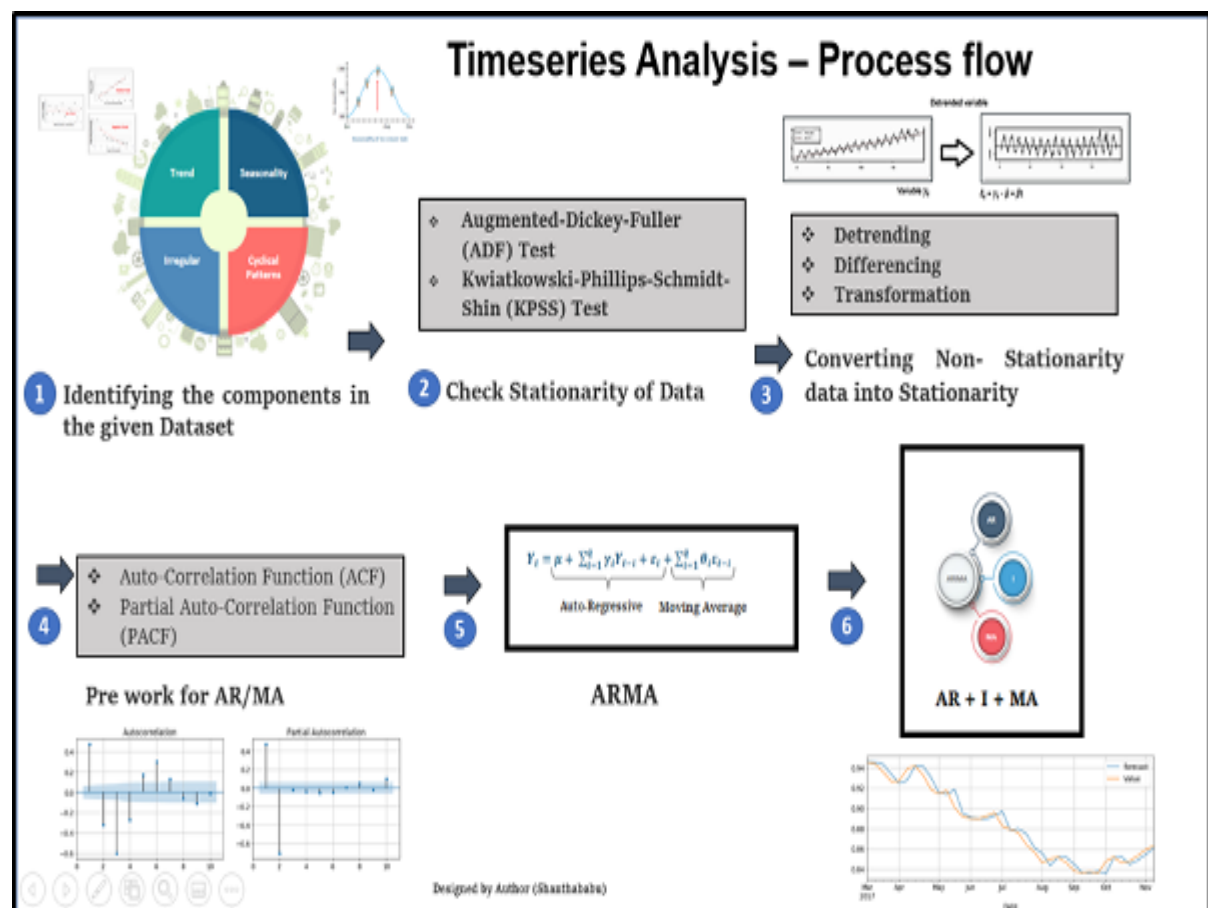
## Understand the signature of ARIMA

- p==> lag order => No of lag observations.
- d==> degree of differencing => No of times that the raw observations are differenced.
- q==>order of moving average => the size of the moving average window

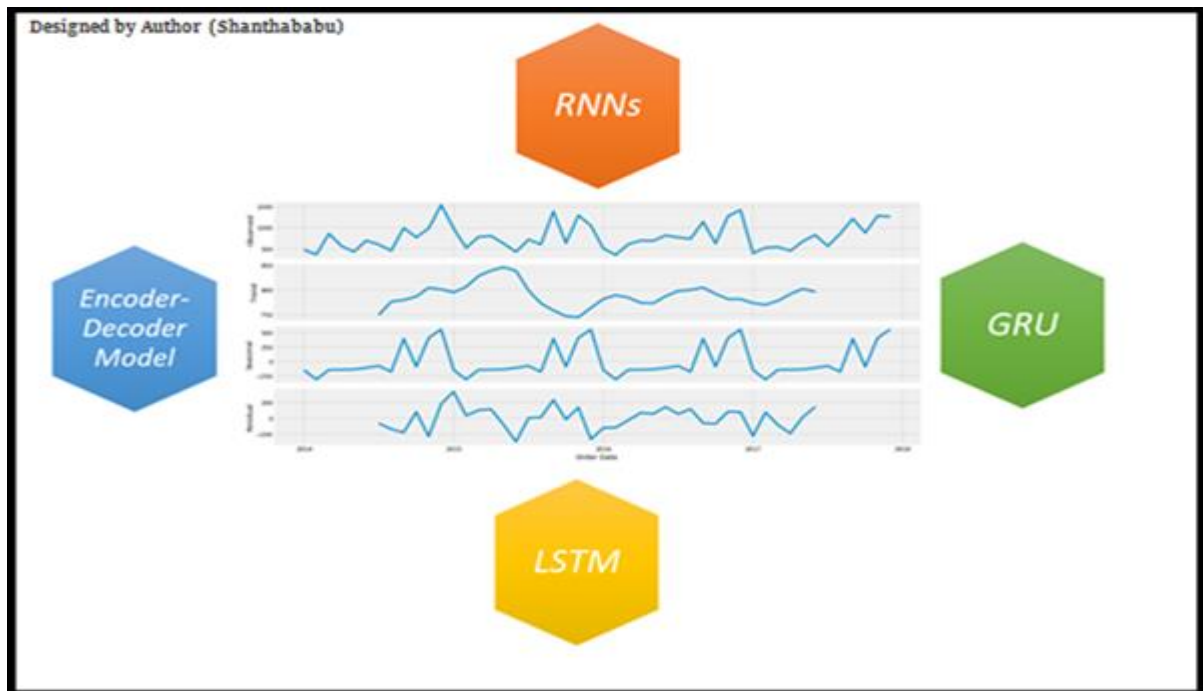
## Implementation Steps for ARIMA

- Plot a time series format
- Difference to make stationary on mean by removing the trend
- Make stationary by applying log transform.
- Difference log transform to make as stationary on both statistic mean and variance
- Plot ACF & PACF, and identify the potential AR and MA model
- Discovery of best fit ARIMA model
- Forecast/Predict the value using the best fit ARIMA model
- Plot ACF & PACF for residuals of the ARIMA model, and ensure no more information is left.

## Process Flow (Re-Gap)



In recent years, the use of Deep Learning for Time Series Analysis and Forecasting has increased to resolve problem statements that couldn't be handled using Machine Learning techniques. Let's discuss this briefly.

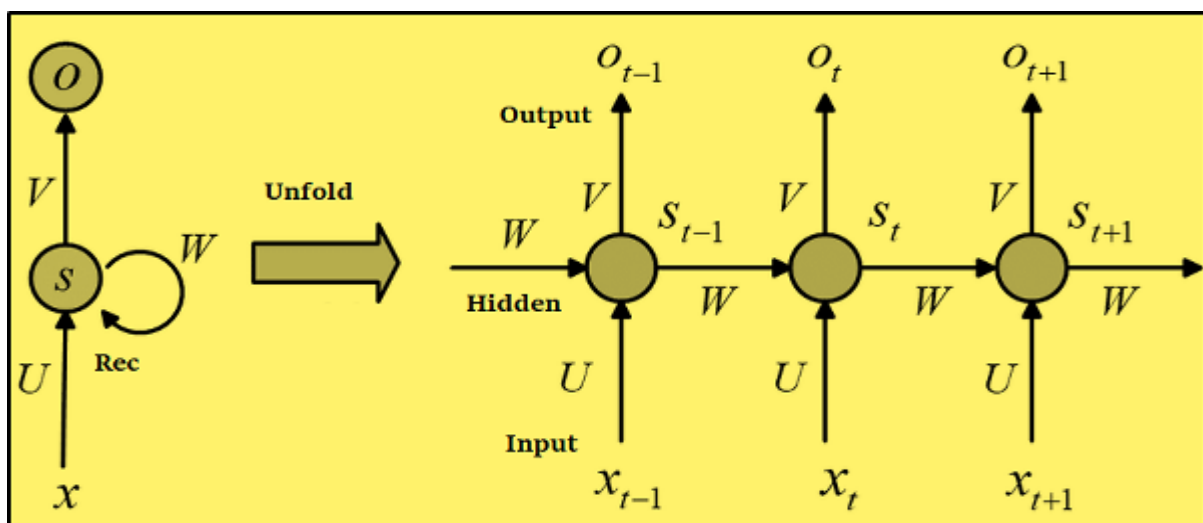


Recurrent Neural Networks (RNN) is the most traditional and accepted architecture fitment for Time-Series forecasting-based problems.

RNN is organized into successive layers and divided into

- Input
- Hidden
- Output

Each layer has equal weight, and every neuron has to be assigned to fixed time steps. Do remember that every one of them is fully connected with a hidden layer (Input and Output) with the same time steps, and the hidden layers are forwarded and time-dependent in direction.



## Components of RNN

- Input: The function vector of  $x(t)$  is the input at time step  $t$ .
- Hidden:
  - The function vector  $h(t)$  is the hidden state at time  $t$ ,
  - This is a kind of memory of the established network;
  - This has been calculated based on the current input  $x(t)$  and the previous-time step's hidden-state  $h(t-1)$ :
- Output: The function vector  $y(t)$  is the output at time step  $t$ .
- Weights : Weights: In the RNNs, the input vector connected to the hidden layer neurons at time  $t$  is by a weight matrix of  $U$  (Please refer to the above picture),

internally weight matrix  $W$  is formed by the hidden layer neurons of time  $t-1$  and  $t+1$ . Following this, the hidden layer with to the output vector  $y(t)$  of time  $t$  by a  $V$  (weight matrix); all the weight matrices  $U$ ,  $W$ , and  $V$  are constant for each time step.

## Advantages of RNN

- It has the special feature that it remembers every piece of information, so RNN is much useful for time series prediction
- Perfect for creating complex patterns from the input time series dataset
- Fast in prediction/forecasting
- Not affected by missing values, so the cleansing process can be limited.

## Disadvantages of RNN

- The big challenge is during the training period
- Expensive computation cost.

## Conclusion

A time series is constructed by data that is measured over time at evenly spaced intervals. I hope this comprehensive guide has helped you all understand the time series, its flow, and how it works. Although the TSA is widely used to handle data science problems, it has certain limitations, such as not supporting missing values. Note that the data points must be linear in their relationship for Time Series Analysis to be done.