Introduction to AI/ML

Learning, Decision Making and ML

Dr. Tamal Ghosh
Department of CSE
Adamas University

Machine Learning –An Introduction

- The aim of machine learning is, making the computers to learn automatically by itself.
- A computer programming that can access data, and some times observe data (external experience) and use it for learning itself (self experience) to take decision is called as machine learning.
- There is no fixed definition for machine learning, and some times we get same definition for both Artificial Intelligence and Machine Learning.

Data in Vector Space Representation

- *Machine Learning is the science of getting computers to learn and act like humans do, and*
- *improve their learning over time in autonomous fashion,*
- *by feeding them data and information in the form of*
- *observations and real-world interactions.*

Types of Machine Learning

- Supervised learning
- Semi supervised learning
- Unsupervised learning
- Reinforcement learning

Supervised Learning

- There are two phases in supervised learning,
- Phase-1 – Model Construction : first train the machine with sample data which are labelled and error free data. i.e. some data are already tagged with correct data.
- The model is represented as classification rules, decision trees, or mathematical formulae
- Phase-2 – Model Usage: This phase will estimate the accuracy of model.
- The new set of data (Testing data) will be given to machine which are example data, then the machine analyze the training data and produces the output.

Semi-Supervised Learning

- This semi supervised learning will fall between supervised and unsupervised learning, and it also called as weak supervision.
- This learning method uses small amount of labeled data and large amount of unlabeled data.
- In this model, the learning (training) will taken place without having much of labeled training data.

Unsupervised Learning

- This model doesn't require labeled data.
- This model learns to identify the patterns and trends or categorize data without labeling it.
- There is more volume of data available for unsupervised learning, because most data are unlabeled.

Reinforcement Learning

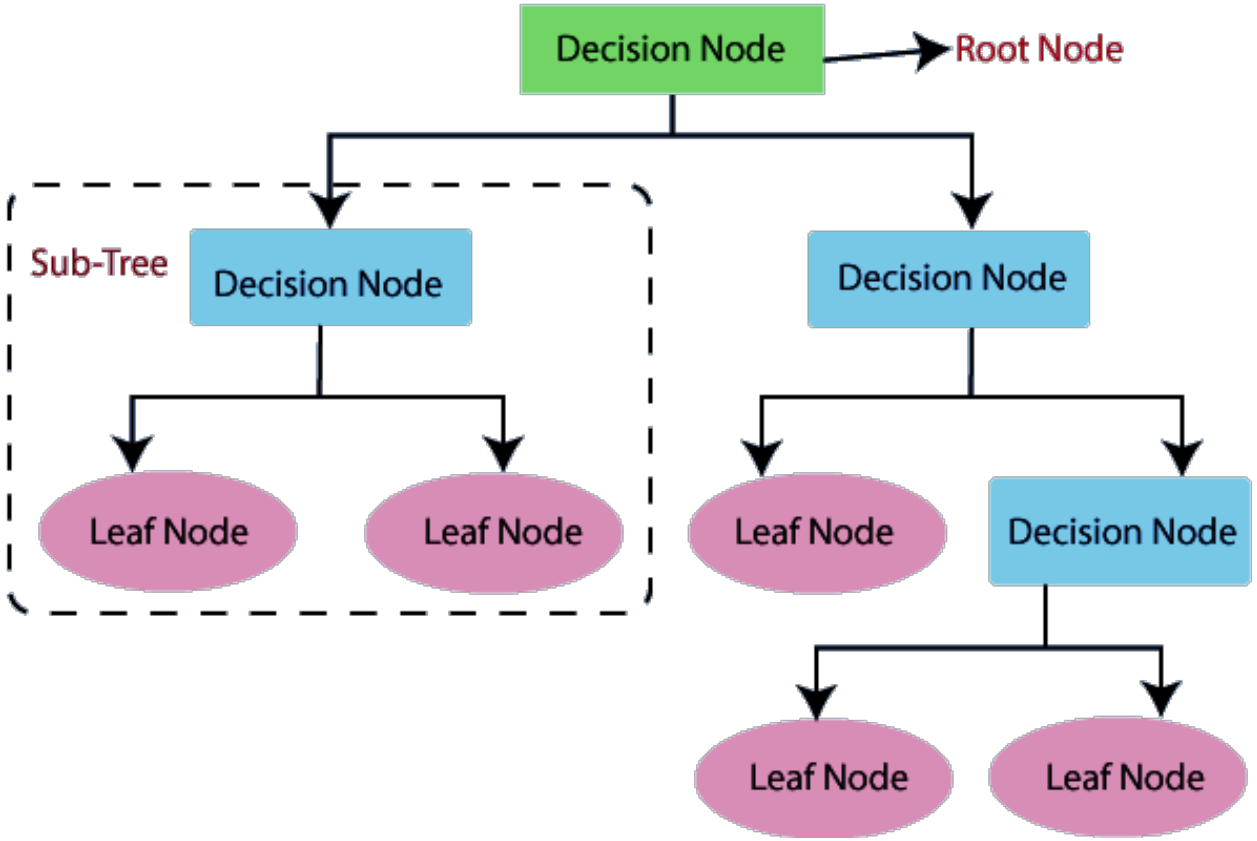
- This model is not like the previous models.
- In reinforcement learning, the reward values are attached to different steps that the model is supposed to go through.
- Hence, the aim here is to accumulate more reward points possible and eventually get to an end goal.
- E.g. Video game.
- To enter into next step, that will earn a reward and taken to next level.

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.**
- The decision tree has two node types, which are the **Decision Node** and **the 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 based on features of the given dataset.
- ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***

Decision Tree: Classification Algorithm

- 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.
- 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 splits the tree into subtrees.
- The diagram at right explains the general structure of a decision tree:



Difference between Fuzzy Set and Crisp Set

- 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.**

Terminologies

- **Root Node:** The 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 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?

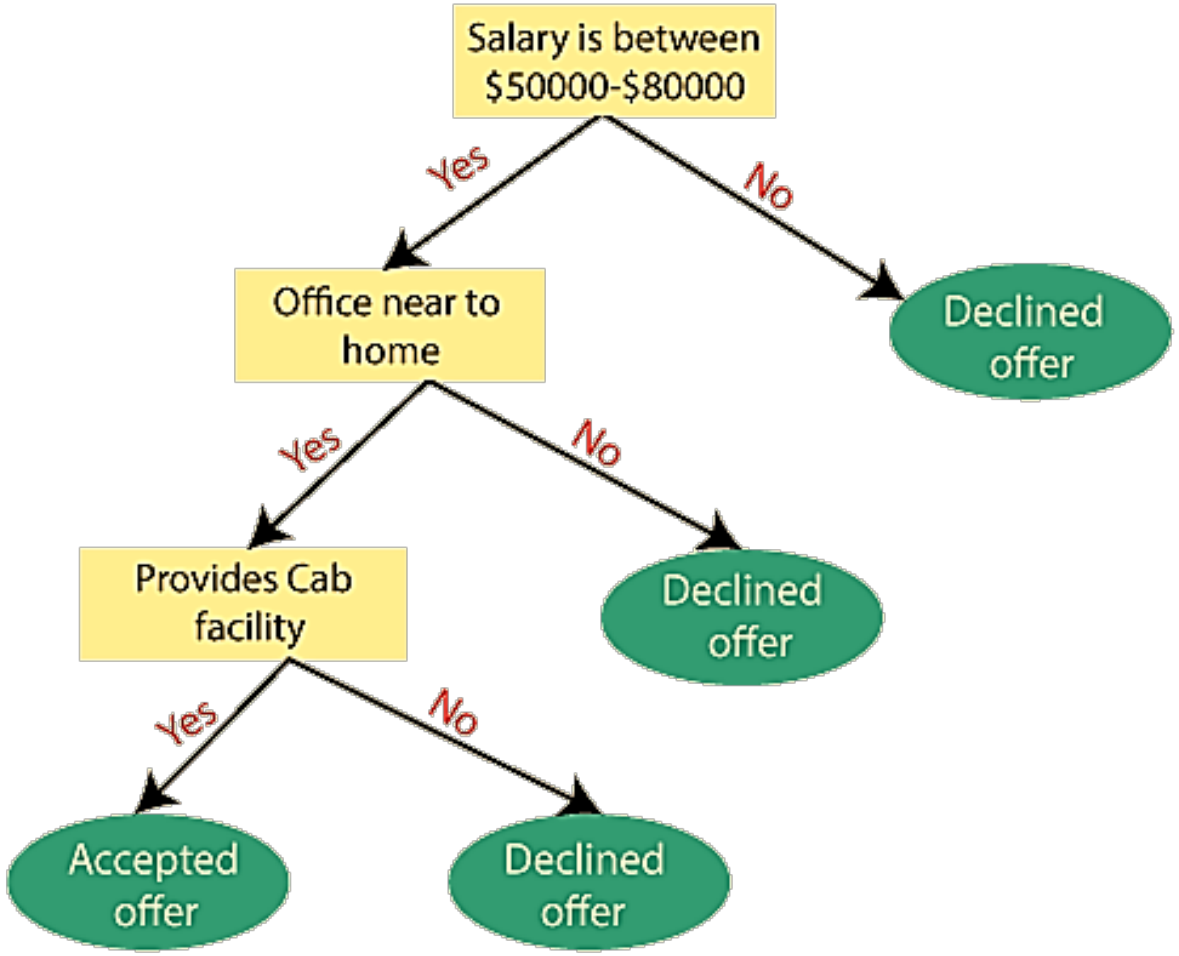
- In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of the root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.
- For the next node, the algorithm again compares the attribute value with the other sub-nodes and moves 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 contain possible values for the best attributes.

Decision Tree Workflow

- **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 call the final node 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 ..



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**

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:

Information Gain= Entropy(S)- (Weighted Avg) *Entropy(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:**

Entropy= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

P(yes)= probability of yes, P(no)= probability of no

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:

$$\text{Gini Index} = 1 - \sum_j P_j^2$$

- There are many algorithms for building a decision tree. They are

1. **CART** (Classification and Regression Trees) — This makes use of Gini impurity as the metric.
2. **ID3** (Iterative Dichotomiser 3) — This uses entropy and information gain as metric.

Pruning: Getting an Optimal Decision Tree

- *Pruning is a process of deleting unnecessary nodes from a tree 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:
- **Pre-Pruning**
- **Post-Pruning**

Pros and Cons of Using Decision Tree

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.

Example

- Consider the dataset to decide whether to play football or not:

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Example

Here we have four independent variables to determine the dependent variable. The independent variables are Outlook, Temperature, Humidity, and Wind. The dependent variable is whether to play football or not.

As the first step, we have to find the parent node for our decision tree. For that follow the steps:

Find the entropy of the class variable.

$E(S) = -[(9/14)\log(9/14) + (5/14)\log(5/14)] = 0.94$

note: Here typically we will take log to base 2. Here total there are 14 yes/no. Out of which 9 yes and 5 no. Based on it we calculated the probability above.

From the above data for Outlook we can arrive at the following table easily

Example

		play		
		yes	no	total
	sunny	3	2	5
Outlook	overcast	4	0	4
	rainy	2	3	5
				14

- ***Now we have to calculate average weighted entropy***. ie, we have found the total of weights of each feature multiplied by probabilities.
- E(S, outlook) = (5/14)*E(3,2) + (4/14)*E(4,0) + (5/14)*E(2,3) = (5/14)(-(3/5)log(3/5)-(2/5)log(2/5))+ (4/14)(0) + (5/14)((2/5)log(2/5)-(3/5)log(3/5)) = 0.693
- ***The next step is to find the information gain***. It is the difference between parent entropy and average weighted entropy we found above.
- IG(S, outlook) = 0.94 - 0.693 = 0.247

Example

- Similarly find Information gain for Temperature, Humidity, and Windy.
- IG(S, Temperature) = 0.940 - 0.911 = 0.029
- IG(S, Humidity) = 0.940 - 0.788 = 0.152
- IG(S, Windy) = 0.940 - 0.8932 = 0.048
- ***Now select the feature having the largest information gain***. Here it is Outlook. So it forms the first node(root node) of our decision tree.
- Now our data look as follows

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Example…

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Overcast	Hot	High	Weak	Yes
Overcast	Cool	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Rain	Mild	Normal	Weak	Yes
Rain	Mild	High	Strong	No

- Since overcast contains only examples of class ‘Yes’ we can set it as yes. That means If the outlook is overcast football will be played. Now our decision tree looks as follows.

Example

The next step is to find the next node in our decision tree. Now we will find one under sunny. We have to determine which of the following Temperature, Humidity or Wind has higher information gain.

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Calculate parent entropy E(sunny)

E(sunny) = (-(3/5)log(3/5)-(2/5)log(2/5)) = 0.971.

Now Calculate the information gain of Temperature. IG(sunny, Temperature)

Example

		play		
		yes	no	total
	hot	0	2	2
Temperature	cool	1	1	2
	mild	1	0	1
				5

E(sunny, Temperature) = (2/5)*E(0,2) + (2/5)*E(1,1) + (1/5)*E(1,0)=2/5=0.4

Now calculate information gain.

IG(sunny, Temperature) = 0.971–0.4 =0.571

Similarly we get

IG(sunny, Humidity) = 0.971

IG(sunny, Windy) = 0.020

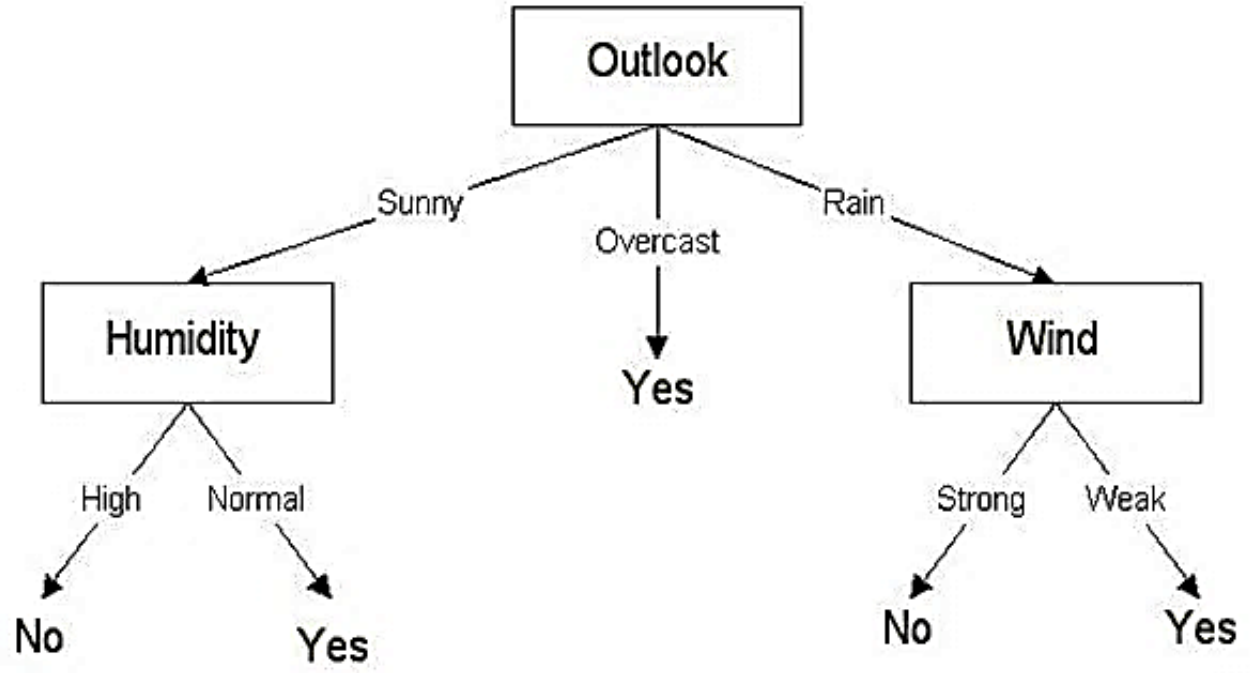
Here IG(sunny, Humidity) is the largest value.

So Humidity is the node that comes under sunny.

	play	
Humidity	yes	no
high	0	3
normal	2	0

Example…

- For humidity from the above table, we can say that play will occur if humidity is normal and will not occur if it is high. Similarly, find the nodes under rainy.
- ***Note: A branch with entropy more than 0 needs further splitting.***
- Finally, our decision tree will look like:



The use of the CART algorithm follows the very same procedure, except it uses Gini impurity rather than entropy.

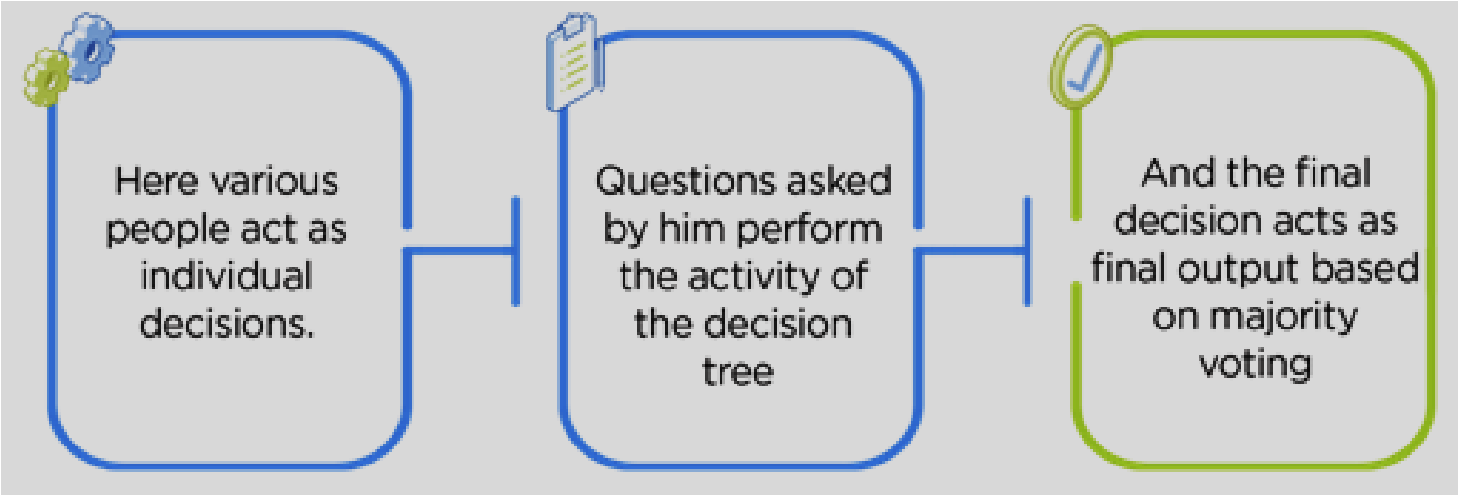
RANDOM FOREST

What is Random Forest Algorithm?

- A Random Forest is like a group decision-making team in machine learning. It combines the opinions of many “trees” (individual models) to make better predictions, creating a more robust and accurate overall model.
- The Random Forest Algorithm is widespread, and popular for its user-friendly nature and adaptability, enabling it to tackle both classification and regression problems effectively. The algorithm’s strength lies in its ability to handle complex datasets and mitigate overfitting, making it a valuable tool for various predictive tasks in machine learning.
- One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables, as in the case of regression, and categorical variables, as in the case of classification. It performs better for classification and regression tasks. In this tutorial, we will understand the working of random forests and implement random forests on a classification task.

Real-Life Analogy of Random Forest

- Suppose a student named X wants to choose a course after his 10+2, and he is confused about the choice of course based on his skill set. So he decides to consult various people like his cousins, teachers, parents, degree students, and working people. He asks them varied questions like why he should choose, job opportunities with that course, course fee, etc. Finally, after consulting various people about the course he decides to take the course suggested by most people.



Working of Random Forest Algorithm

- Before understanding the workings of the random forest algorithm in machine learning, we must look into the ensemble learning technique. ***Ensemble*** simply means combining multiple models. Thus a collection of models is used to make predictions rather than an individual model.
- Ensemble uses two types of methods:

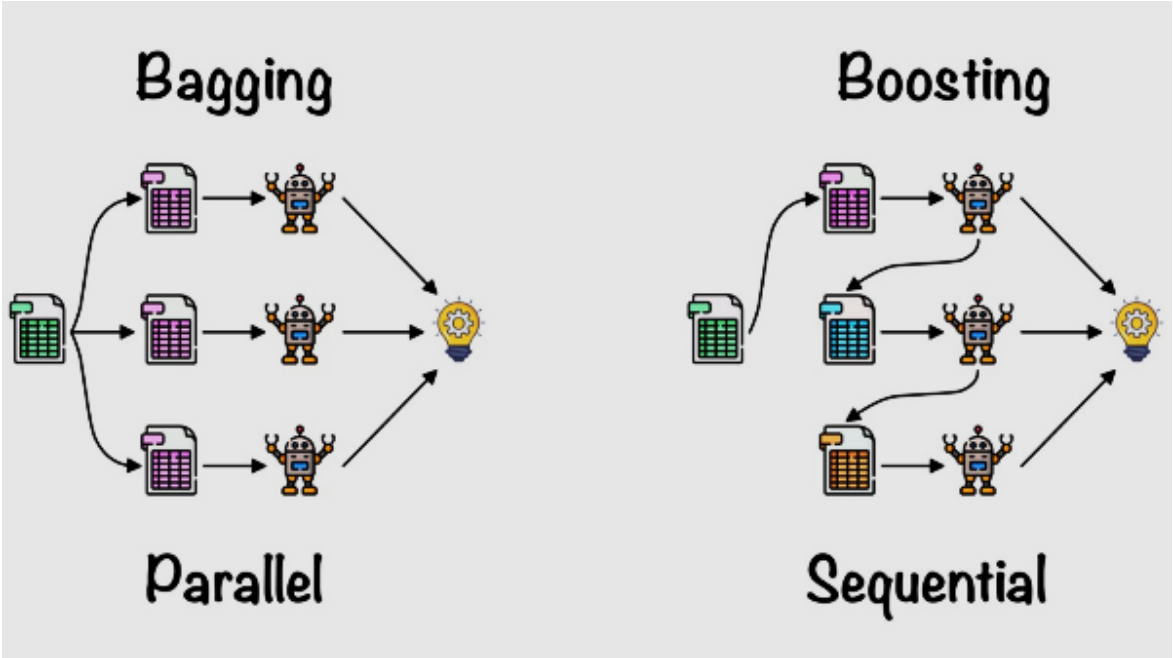
Bagging

- It creates a different training subset from sample training data with replacement and the final output is based on majority voting. For example, Random Forest.

Boosting

- It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.

Ensemble Learning Technique



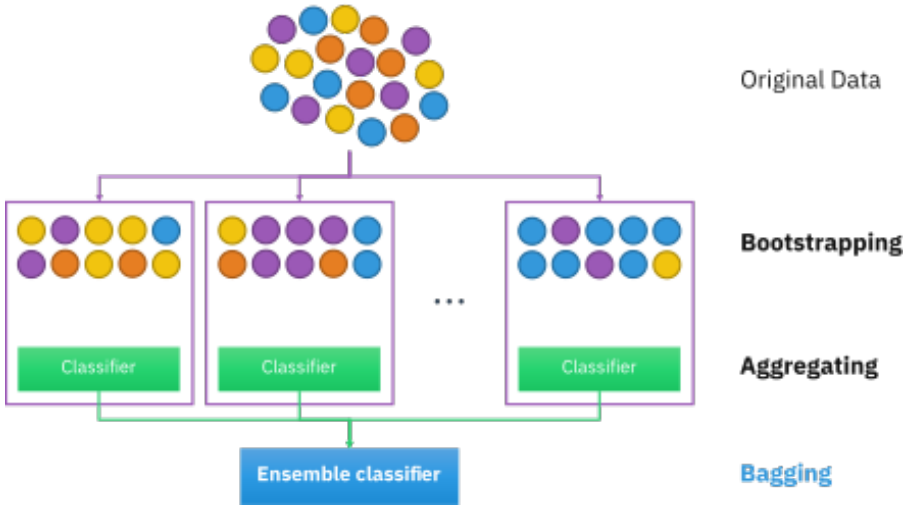
- As mentioned earlier, Random forest works on the Bagging principle. Now let’s dive in and understand bagging in detail.

Bagging

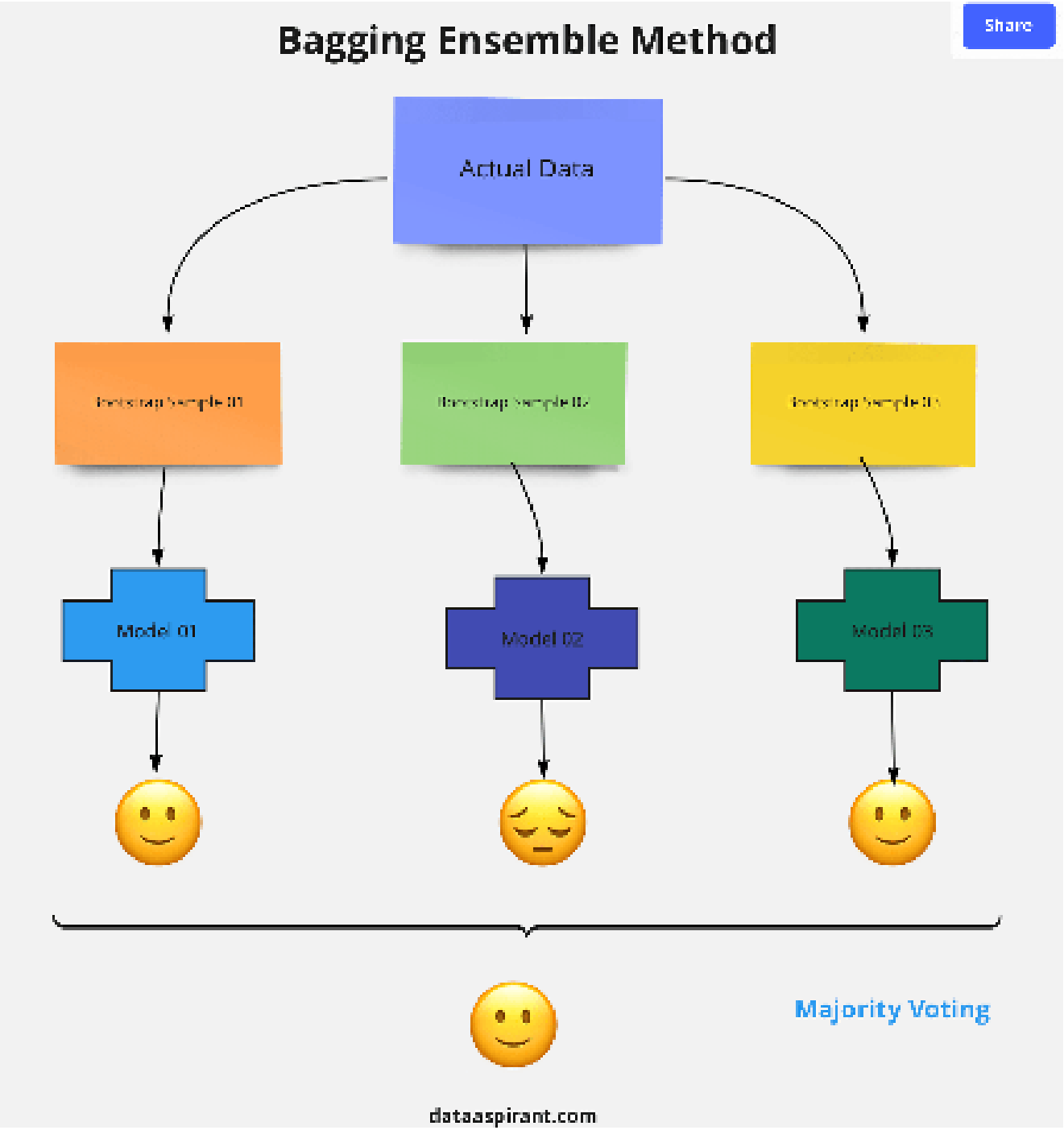
- Bagging, also known as Bootstrap Aggregation, serves as the ensemble technique in the Random Forest algorithm. Here are the steps involved in Bagging:

1. **Selection of Subset**: Bagging starts by choosing a random sample from the entire dataset.
2. **Bootstrap Sampling**: Each model is then created from these samples, called Bootstrap Samples, which are taken from the original data with replacement. This process is known as row sampling.
3. **Bootstrapping**: The step of row sampling with replacement is referred to as bootstrapping.
4. **Independent Model Training**: Each model is trained independently on its corresponding Bootstrap Sample. This training process generates results for each model.
5. **Majority Voting**: The final output is determined by combining the results of all models through majority voting. The most commonly predicted outcome among the models is selected.
6. **Aggregation**: This step, which involves combining all the results and generating the final output based on majority voting, is known as aggregation.

Bagging…



- Now let's look at an example by breaking it down with the help of the following figure. Here the bootstrap sample is taken from actual data (Bootstrap sample 01, Bootstrap sample 02, and Bootstrap sample 03) with a replacement which means there is a high possibility that each sample won't contain unique data. The model (Model 01, Model 02, and Model 03) obtained from this bootstrap sample is trained independently. Each model generates results as shown. Now the Happy emoji has a majority when compared to the Sad emoji. Thus based on majority voting final output is obtained as Happy emoji.


Bagging Ensemble Method
Share
Actual Data
Model 01
Model 02
Majority Voting
dataaspirant.com

Boosting

- Boosting is one of the techniques that use the concept of ensemble learning. A boosting algorithm combines multiple simple models (also known as weak learners or base estimators) to generate the final output. It is done by building a model by using weak models in series.

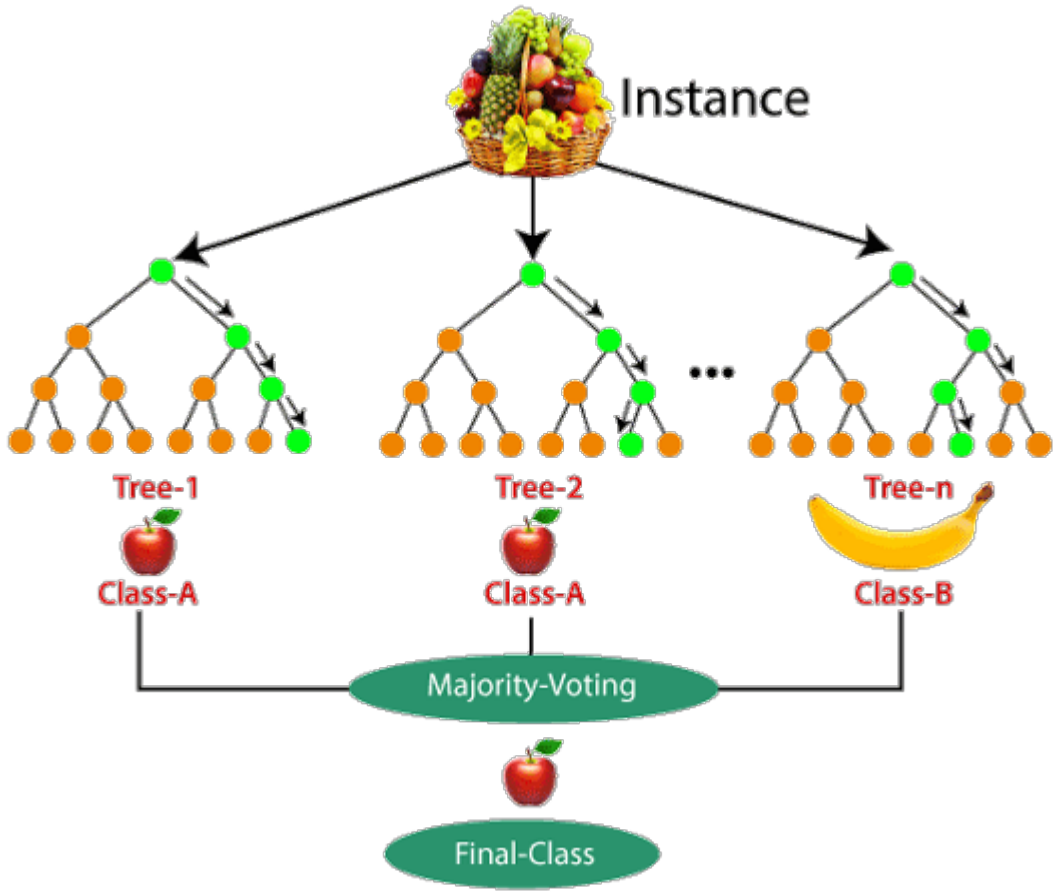
- There are several boosting algorithms; AdaBoost was the first really successful boosting algorithm that was developed for the purpose of binary classification. AdaBoost is an abbreviation for Adaptive Boosting and is a prevalent boosting technique that combines multiple “weak classifiers” into a single “strong classifier.”

Steps Involved in Random Forest Algorithm

- Step 1: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.
- Step 2: Individual decision trees are constructed for each sample.
- Step 3: Each decision tree will generate an output.
- Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression, respectively.

Example

- Consider the fruit basket as the data as shown in the figure below. Now n number of samples are taken from the fruit basket, and an individual decision tree is constructed for each sample. Each decision tree will generate an output, as shown in the figure. The final output is considered based on majority voting. In the below figure, you can see that the majority decision tree gives output as an apple when compared to a banana, so the final output is taken as an apple.



Important Features of Random Forest

- **Diversity:** Not all attributes/variables/features are considered while making an individual tree; each tree is different.
- **Immune to the curse of dimensionality:** Since each tree does not consider all the features, the feature space is reduced.
- **Parallelization:** Each tree is created independently out of different data and attributes. This means we can fully use the CPU to build random forests.
- **Train-Test split:** In a random forest, we don't have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
- **Stability:** Stability arises because the result is based on majority voting/ averaging.

Difference Between Decision Tree and Random Forest

Decision trees	Random Forest
1. Decision trees normally suffer from the problem of overfitting if it's allowed to grow without any control.	1. Random forests are created from subsets of data, and the final output is based on average or majority ranking; hence the problem of overfitting is taken care of.
2. A single decision tree is faster in computation.	2. It is comparatively slower.
3. When a data set with features is taken as input by a decision tree, it will formulate some rules to make predictions.	3. Random forest randomly selects observations, builds a decision tree, and takes the average result. It doesn't use any set of formulas.

Important Hyperparameters in Random Forest

- Hyperparameters are used in random forests to either enhance the performance and predictive power of models or to make the model faster.

Hyperparameters to Increase the Predictive Power

- **n_estimators:** Number of trees the algorithm builds before averaging the predictions.
- **max_features:** Maximum number of features random forest considers splitting a node.
- **mini_sample_leaf:** Determines the minimum number of leaves required to split an internal node.
- **criterion:** How to split the node in each tree? (Entropy/Gini impurity/Log Loss)
- **max_leaf_nodes:** Maximum leaf nodes in each tree

Hyperparameters…

Hyperparameters to Increase the Speed

- ***n_jobs:*** it tells the engine how many processors it is allowed to use. If the value is 1, it can use only one processor, but if the value is -1, there is no limit.
- ***random_state:*** controls randomness of the sample. The model will always produce the same results if it has a definite value of random state and has been given the same hyperparameters and training data.
- ***oob_score:*** *OOB* means out of the bag. It is a random forest cross-validation method. In this, one-third of the sample is not used to train the data; instead used to evaluate its performance. These samples are called out-of-bag samples.

Advantages and Disadvantages of Random Forest

Advantages

- It can be used in classification and regression problems.
- It solves the problem of overfitting as output is based on majority voting or averaging.
- It performs well even if the data contains null/missing values.
- Each decision tree created is independent of the other; thus, it shows the property of parallelization.
- It is highly stable as the average answers given by a large number of trees are taken.
- It maintains diversity as all the attributes are not considered while making each decision tree though it is not true in all cases.
- It is immune to the curse of dimensionality. Since each tree does not consider all the attributes, feature space is reduced.
- We don’t have to segregate data into train and test as there will always be 30% of the data, which is not seen by the decision tree made out of bootstrap.

Advantages and Disadvantages of Random Forest

Disadvantages

- Random forest is highly complex compared to decision trees, where decisions can be made by following the path of the tree.
- Training time is more than other models due to its complexity. Whenever it has to make a prediction, each decision tree has to generate output for the given input data.

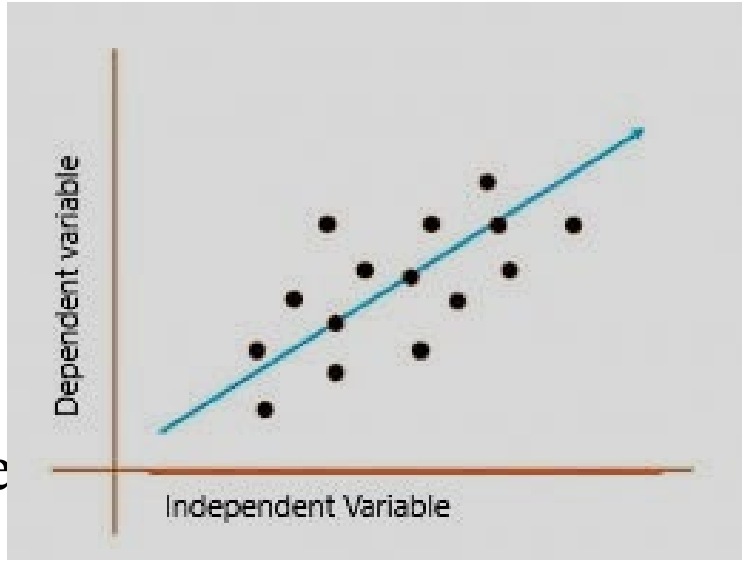
REGRESSION

Linear Regression

- Linear regression predicts the relationship between two variables by assuming a linear connection between the independent and dependent variables.
- It seeks the optimal line that minimizes the sum of squared differences between predicted and actual values.
- Applied in various domains like economics and finance, this method analyzes and forecasts data trends.
- It can extend to multiple linear regression involving several independent variables and logistic regression, suitable for binary classification problems
- Linear regression shows the linear relationship between the independent(predictor) variable i.e. X-axis and the dependent(output) variable i.e. Y-axis, called linear regression.

Simple Linear Regression

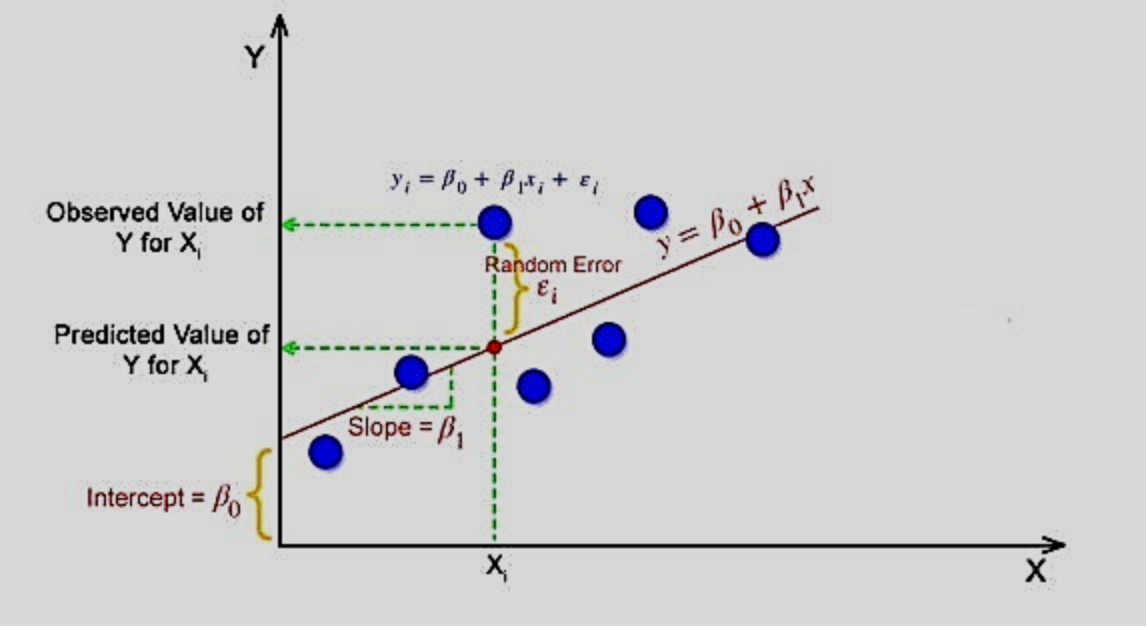
- If there is a single input variable X(independent variable), such linear regression is simple linear regression.
- In a simple linear regression, there is one independent variable and one dependent variable.
- The model estimates the slope and intercept of the line of best fit, which represents the relationship between the variables.
- The slope represents the change in the dependent variable for each unit change in the independent variable, while the intercept represents the predicted value of the dependent variable when the independent variable is zero.



Simple Regression Calculation

- To calculate best-fit line linear regression uses a traditional slope-intercept form which is given below,
- $Y = \beta_0 + \beta_1 X$
- where Y_i = Dependent variable, β_0 = constant/Intercept, β_1 = Slope/Intercept, X_i = Independent variable.

This algorithm explains the linear relationship between the dependent (output) variable y and the independent(predictor) variable X using a straight line $Y = \beta_0 + \beta_1 X$.



Simple Regression Calculation

- The goal of the linear regression algorithm is to get the **best values for β_0 and β_1** to find the best-fit line. The best-fit line is a line that has the least error which means the error between predicted values and actual values should be minimum.

Random Error(Residuals)

- In regression, the difference between the observed value of the dependent variable(y_i) and the predicted value(**predicted**) is called the residuals.
- $\varepsilon_i = y_{predicted} - y_i$
- **where** $y_{predicted} = \beta_0 + \beta_1 X_i$

What is the best-fit line?

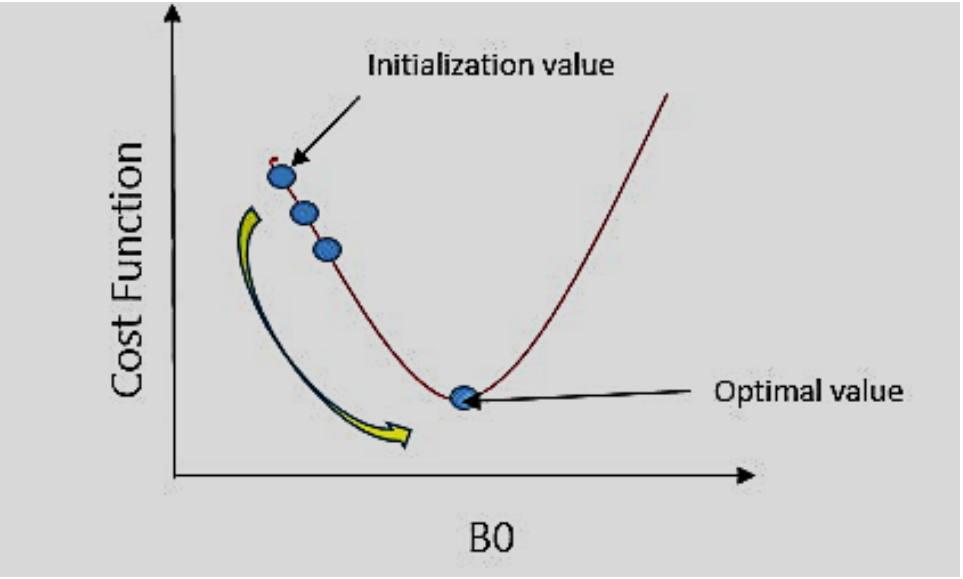
- In simple terms, the best-fit line is a line that fits the given scatter plot in the best way. Mathematically, the best-fit line is obtained by minimizing the Residual Sum of Squares(RSS).

Cost Function for Linear Regression

- The cost function helps to work out the optimal values for β_0 and β_1, which provides the best-fit line for the data points.
- In Linear Regression, generally **Mean Squared Error (MSE)** cost function is used, which is the average squared error that occurred between the $y_{predicted}$ and y_i.
- We calculate MSE using the simple linear equation y=mx+b:
- Using the $$MSE = \frac{1}{N}\sum_{i=1}^{n}(y_i - (B1 x_i + B0))^2$$ the MSE value settles at the minima. These parameters can be determined using the gradient descent method such that the value for the cost function is minimum.

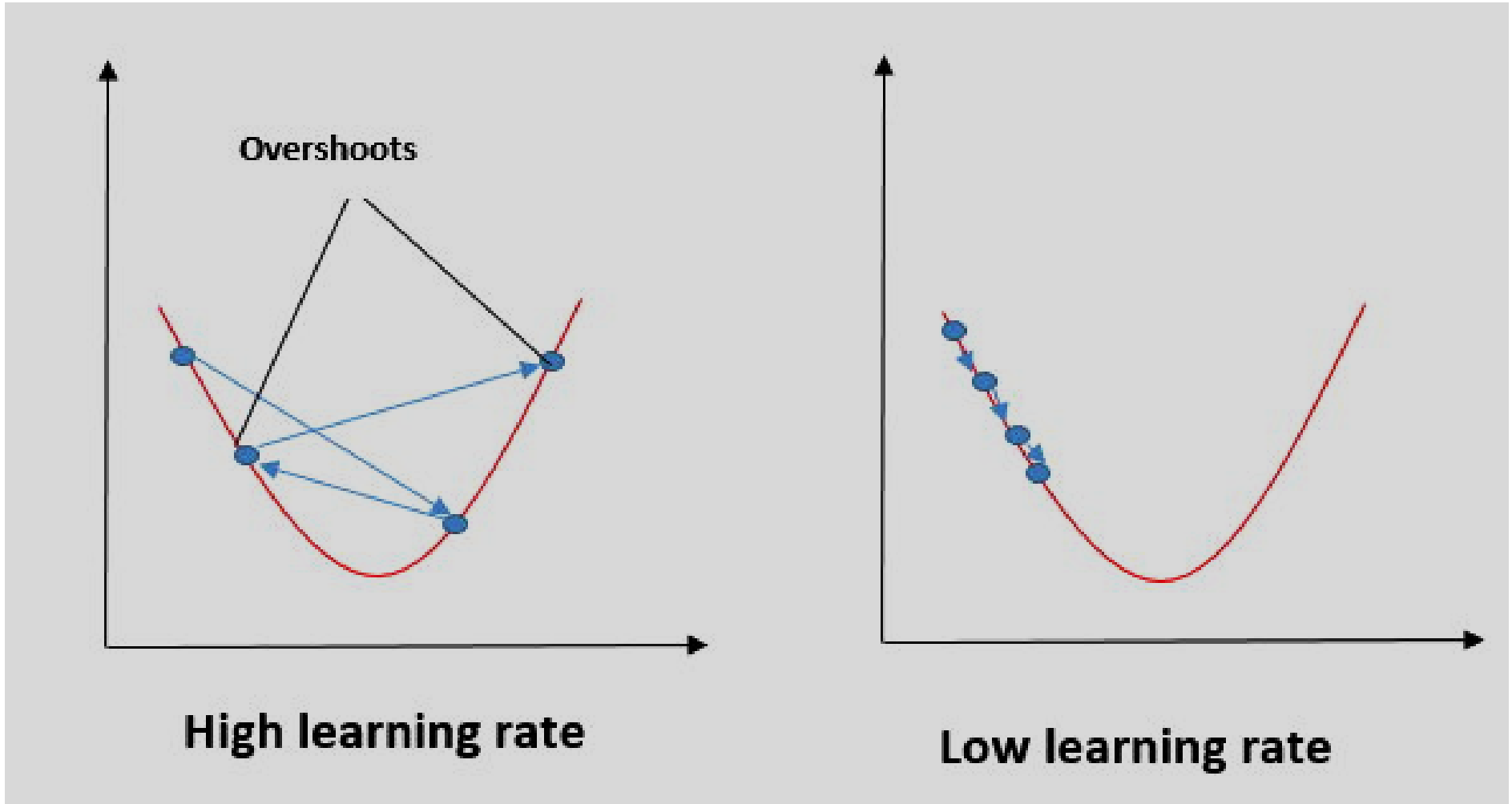
Gradient Descent for Linear Regression

- Gradient Descent is one of the optimization algorithms that optimize the cost function(objective function) to reach the optimal minimal solution. To find the optimum solution we need to reduce the cost function(MSE) for all data points. This is done by updating the values of B_0 and B_1 iteratively until we get an optimal solution.
- A regression model optimizes the gradient descent algorithm to update the coefficients of the line by reducing the cost function by randomly selecting coefficient values and then iteratively updating the values to reach the minimum cost function.



Gradient Descent Example

- Imagine a U-shaped pit and you are standing at the uppermost point in the pit, and your motive is to reach the bottom of the pit.
- Suppose there is a treasure at the bottom of the pit, and you can only take a discrete number of steps to reach the bottom. If you opted to take one step at a time, you would get to the bottom of the pit in the end but, this would take a longer time. If you decide to take larger steps each time, you may achieve the bottom sooner but, there's a probability that you could overshoot the bottom of the pit and not even near the bottom.
- In the gradient descent algorithm, the number of steps you're taking can be considered as the **learning rate**, and this decides how fast the algorithm **converges** to the minima.


Overshoots
High learning rate
Low learning rate

Gradient Descent Example

- To update B_0 and B_1, we take gradients from the cost function. To find these gradients, we take partial derivatives for B_0 and B_1.
- We need to minimize the cost function J. One of the ways to achieve this is to apply the batch gradient descent algorithm, the values are updated in each iteration. (The last two equations show the updating of values)
- The partial derivatives are the gradients, and they are used to update the values of B_0 and B_1. Alpha is the learning rate.

$$J = \frac{1}{n}\sum_{i=1}^{n}(B_0 + B_1 \cdot x_i - y_i)^2$$

$$\frac{\partial J}{\partial B_0} = \frac{2}{n}\sum_{i=1}^{n}(B_0 + B_1 \cdot x_i - y_i)$$

$$\frac{\partial J}{\partial B_1} = \frac{2}{n}\sum_{i=1}^{n}(B_0 + B_1 \cdot x_i - y_i) \cdot x_i$$

$$J = \frac{1}{n}\sum_{i=1}^{n}(B_0 + B_1 \cdot x_i - y_i)^2$$

$$\frac{\partial J}{\partial B_0} = \frac{2}{n}\sum_{i=1}^{n}(B_0 + B_1 \cdot x_i - y_i)$$

$$\frac{\partial J}{\partial B_1} = \frac{2}{n}\sum_{i=1}^{n}(B_0 + B_1 \cdot x_i - y_i) \cdot x_i$$

$$B_0 = B_0 - \alpha \cdot \frac{2}{n}\sum_{i=1}^{n}(pred_i - y_i)$$

$$B_1 = B_1 - \alpha \cdot \frac{2}{n}\sum_{i=1}^{n}(pred_i - y_i) \cdot x_i$$

Evaluation Metrics for Linear Regression

- The strength of any linear regression model can be assessed using various evaluation metrics. These evaluation metrics usually provide a measure of how well the observed outputs are being generated by the model.
- The most used metrics are,

1. Coefficient of Determination or R-squared (R2)
2. Root Mean Squared Error (RSME) and Residual Standard Error (RSE)

Coefficient of Determination or R-squared (R2)

- R-squared is a number that explains the amount of variation that is explained/captured by the developed model. It always ranges between 0 and 1 . Overall, the higher the value of R-squared, the better the model fits the data.
- Mathematically it can be represented as,
- $$R^2 = 1 – (RSS/TSS)$$

Metrics…

- **Residual sum of Squares (RSS)** is defined as the sum of squares of the residual for each data point in the plot/data. It is the measure of the difference between the expected and the actual observed output.

$$\text{RSS} = \sum_{i=1}^{n} (y_i - b_0 - b_1 x_i)^2$$

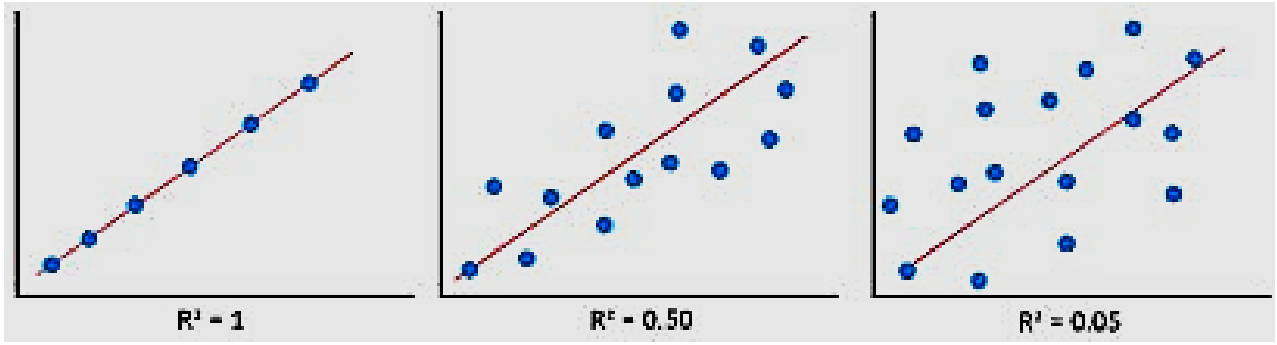
- **Total Sum of Squares (TSS)** is defined as the sum of errors of the data points from the mean of the response variable. Mathematically TSS is,

$$TSS = \sum (y_i - \bar{y}_i)^2$$

- where y hat is the mean of the sample data points.

Metrics…

- The significance of R-squared is shown by the following figures,



- The **Root Mean Squared Error** is the square root of the variance of the residuals. It specifies the absolute fit of the model to the data i.e. how close the observed data points are to the predicted values. Mathematically it can be represented as,

$$RMSE = \sqrt{\frac{RSS}{n}} = \sqrt{\sum_{i=1}^{n} \left(y_i^{Actual} - y_i^{Predicted}\right)^2 / n}$$

Metrics…

- To make this estimate unbiased, one has to divide the sum of the squared residuals by the **degrees of freedom** rather than the total number of data points in the model. This term is then called the **Residual Standard Error(RSE)**. Mathematically it can be represented as,

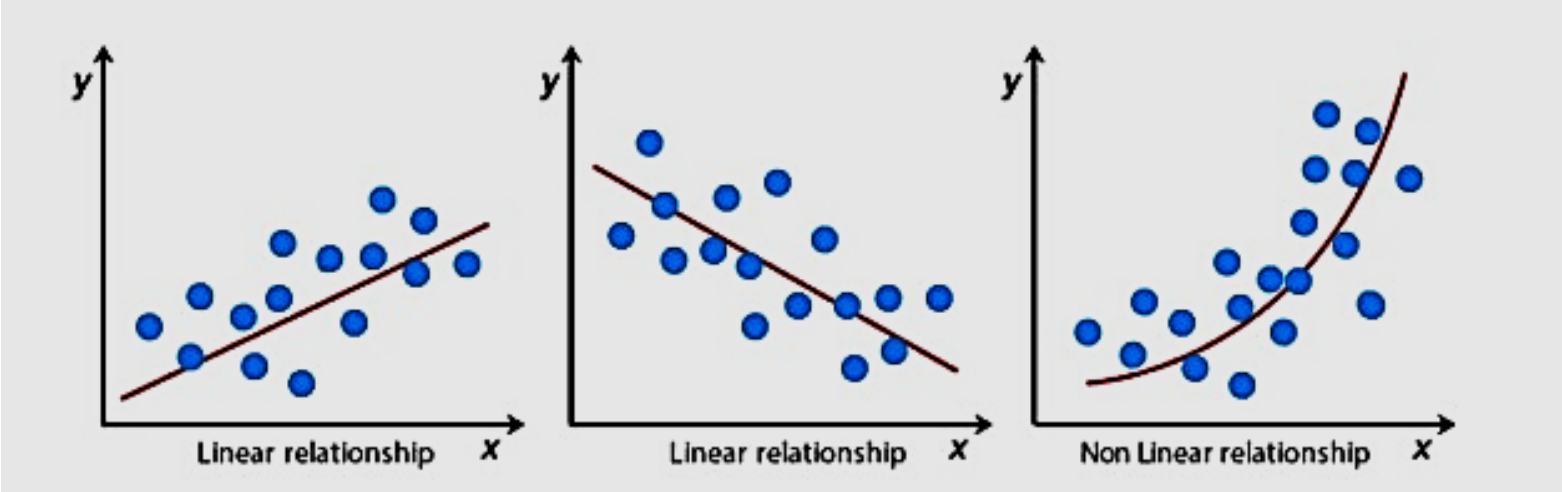
$$RSE = \sqrt{\frac{RSS}{df}} = \sqrt{\sum_{i=1}^{n} \left(y_i^{Actual} - y_i^{Predicted}\right)^2 / (n-2)}$$

- R-squared is a better measure than RSME. Because the value of Root Mean Squared Error depends on the units of the variables (i.e. it is not a normalized measure), it can change with the change in the unit of the variables.

Assumptions of Linear Regression

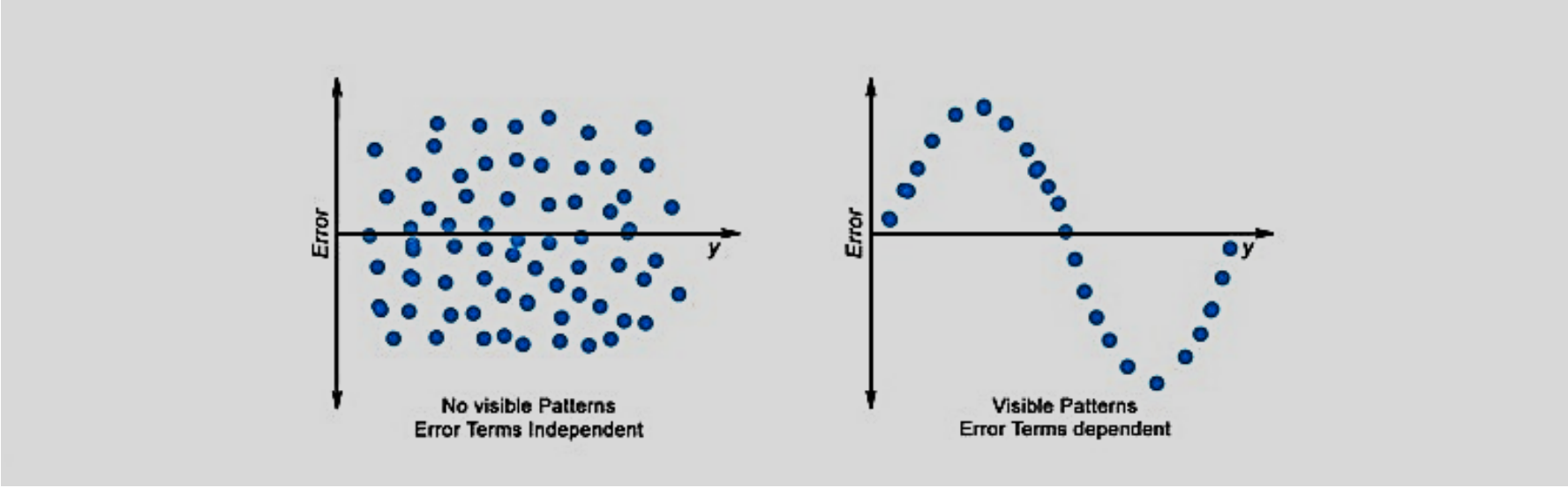
- Regression is a parametric approach, which means that it makes assumptions about the data for the purpose of analysis. For successful regression analysis, it's essential to validate the following assumptions.

1. **Linearity of residuals**: There needs to be a linear relationship between the dependent variable and independent variable(s).



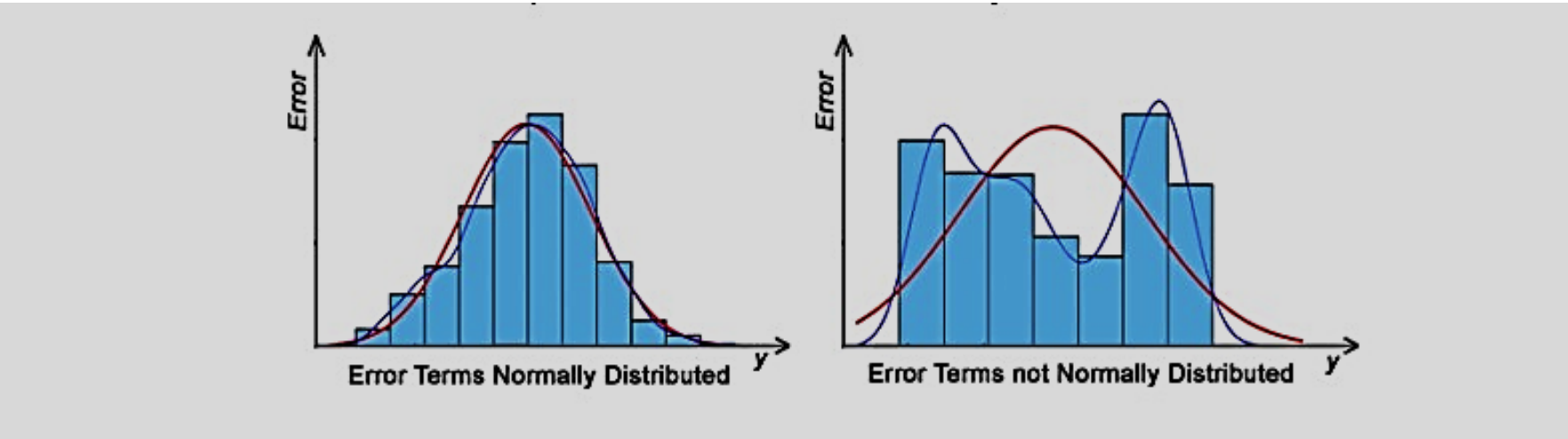
Assumptions…

- . **Independence of residuals:** The error terms should not be dependent on one another (like in time-series data wherein the next value is dependent on the previous one). There should be no correlation between the residual terms. The absence of this phenomenon is known as **Autocorrelation.**
- There should not be any visible patterns in the error terms.



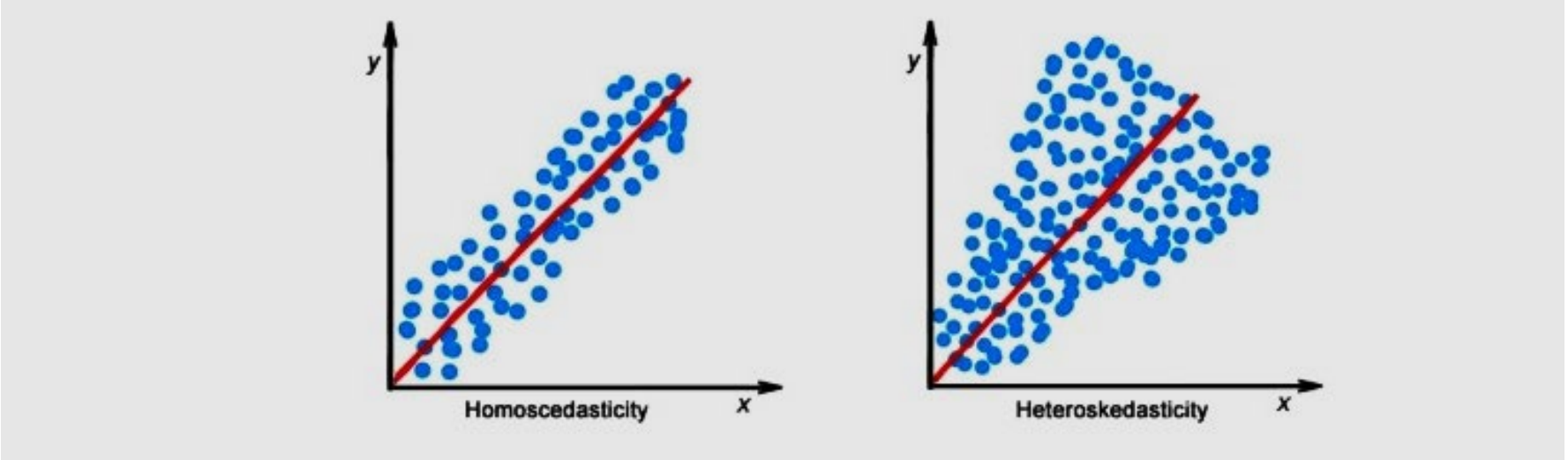
Assumptions…

- **<u>Normal distribution of residuals:</u>** The mean of residuals should follow a normal distribution with a mean equal to zero or close to zero. This is done in order to check whether the selected line is actually the line of best fit or not.
  If the error terms are non-normally distributed, suggests that there are a few unusual data points that must be studied closely to make a better model.



Assumptions…

- **The equal variance of residuals:** The error terms must have constant variance. This phenomenon is known as **Homoscedasticity**.
- The presence of non-constant variance in the error terms is referred to as **Heteroscedasticity**. Generally, non-constant variance arises in the presence of *outliers or extreme leverage values*.



Hypothesis in Linear Regression

- Once you have fitted a straight line on the data, you need to ask, "**Is this straight line a significant fit for the data?**" Or "I**s the beta coefficient explain the variance in the data plotted?"** And here comes the idea of **hypothesis testing** on the beta coefficient. The Null and Alternate hypotheses in this case are:
- H_0: $B_1 = 0$
- H_A: $B_1 \neq 0$
- To test this hypothesis we use a **t-test,** test statistics for the beta coefficient is given by,

$$t = \frac{m - \mu}{s/\sqrt{n}}$$

Assessing the model fit

- Some other parameters to assess a model are:

1. **t statistic:** It is used to determine the p-value and hence, helps in determining whether the coefficient is significant or not
2. **F statistic**: It is used to assess whether the overall model fit is significant or not. Generally, the higher the value of the F-statistic, the more significant a model turns out to be.

Multiple Linear Regression

- Multiple linear regression is a technique to understand the relationship between a *single* dependent variable and *multiple* independent variables.
- The formulation for multiple linear regression is also similar to simple linear regression with
- the small change that instead of having one beta variable, you will now have betas for all the variables used. The formula is given as:
- $Y = B_0 + B_1X_1 + B_2X_2 + \ldots + B_pX_p + \boldsymbol{\varepsilon}$

Considerations of Multiple Linear Regression

- All the four assumptions made for Simple Linear Regression still hold true for Multiple Linear Regression along with a few new additional assumptions.

1. **Overfitting**: When more and more variables are added to a model, the model may become far too complex and usually ends up memorizing all the data points in the training set. This phenomenon is known as the overfitting of a model. This usually leads to high training accuracy and very low test accuracy.
2. **Multicollinearity**: It is the phenomenon where a model with several independent variables, may have some variables interrelated.
3. **Feature Selection:** With more variables present, selecting the optimal set of predictors from the pool of given features (many of which might be redundant) becomes an important task for building a relevant and better model.

Multicollinearity

- As multicollinearity makes it difficult to find out which variable is contributing towards the prediction of the response variable, it leads one to conclude incorrectly, the effects of a variable on the target variable. Though it does not affect the precision of the predictions, it is essential to properly detect and deal with the multicollinearity present in the model, as random removal of any of these correlated variables from the model causes the coefficient values to swing wildly and even change signs.
- Multicollinearity can be detected using the following methods.

1. **Pairwise Correlations:** Checking the pairwise correlations between different pairs of independent variables can throw useful insights into detecting multicollinearity.

Multicollinearity…

- **Variance Inflation Factor (VIF):** Pairwise correlations may not always be useful as it is possible that just one variable might not be able to completely explain some other variable but some of the variables combined could be ready to do this. Thus, to check these sorts of relations between variables, one can use VIF. VIF explains the relationship of one independent variable with all the other independent variables. VIF is given by,

$$VIF = \frac{1}{1 - R^2}$$

- *where i* refers to the i^{th} variable which is represented as a linear combination of the rest of the independent variables.
- The common heuristic followed for the VIF values is if VIF > 10 then the value is high and it should be dropped. And if the VIF=5 then it may be valid but should be inspected first. If VIF < 5, then it is a good VIF value.

Overfitting and Underfitting in Linear Regression

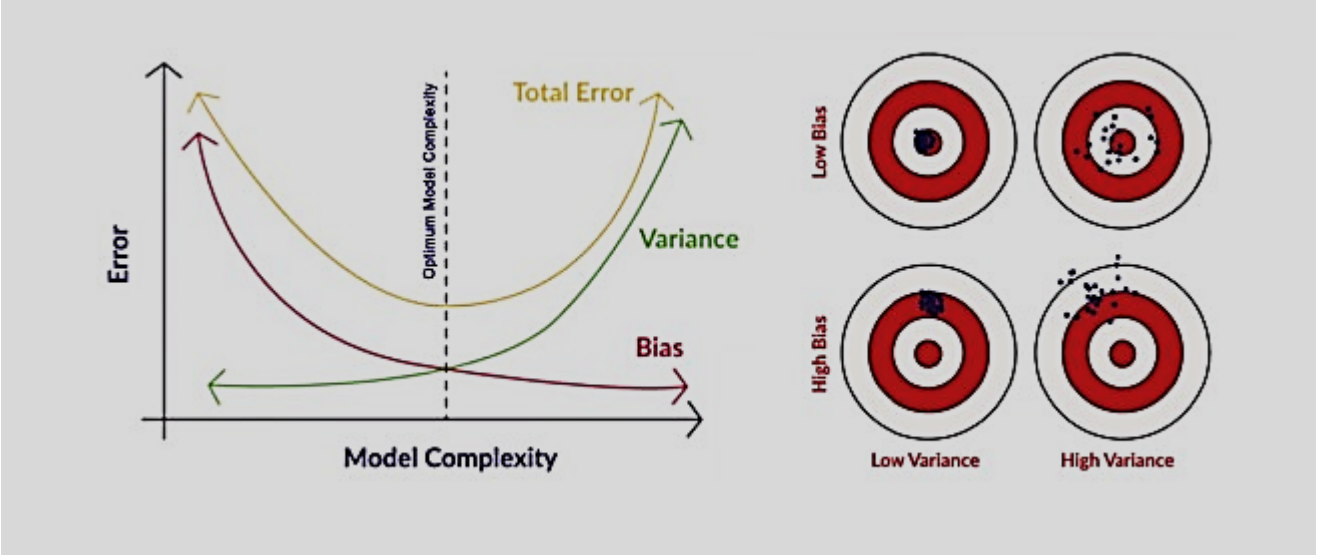
- There have always been situations where a model performs well on training data but not on the test data. While training models on a dataset, overfitting, and underfitting are the most common problems faced by people.
- Before understanding overfitting and underfitting one must know about bias and variance.

Bias:

- Bias is a measure to determine how accurate is the model likely to be on future unseen data. Complex models, assuming there is enough training data available, can do predictions accurately. Whereas the models that are too naive, are very likely to perform badly with respect to predictions. Simply, Bias is errors made by training data.

Overfitting…

- Generally, linear algorithms have a high bias which makes them fast to learn and easier to understand but in general, are less flexible. Implying lower predictive performance on complex problems that fail to meet the expected outcomes.
- **Variance:**
- Variance is the sensitivity of the model towards training data, that is it quantifies how much the model will react when input data is changed.
- Ideally, the model shouldn't change too much from one training dataset to the next training data, which means that the algorithm is good at picking out the hidden underlying patterns between the inputs and the output variables.
- Ideally, a model should have lower variance which means that the model doesn't change drastically after changing the training data(it is generalizable). Having a higher variance will make a model change drastically even on a small change in the training dataset.
- Let's understand what a bias-variance tradeoff is.



- In the pursuit of optimal performance, a supervised machine learning algorithm seeks to strike a balance between low bias and low variance for increased robustness.
- In the realm of machine learning, there exists an inherent relationship between bias and variance, characterized by an inverse correlation.
  - Increased bias leads to reduced variance.
  - Conversely, heightened variance results in diminished bias.

Bias…

- Finding an equilibrium between bias and variance is crucial, and algorithms must navigate this trade-off for optimal outcomes.
- In practice, calculating precise bias and variance error terms is challenging due to the unknown nature of the underlying target function.
- Now, let’s delve into the nuances of overfitting and underfitting.

Overfitting

- When a model learns each and every pattern and noise in the data to such extent that it affects the performance of the model on the unseen future dataset, it is referred to as ***overfitting***. The model fits the data so well that it interprets noise as patterns in the data.
- When a model has low bias and higher variance it ends up memorizing the data and causing overfitting. Overfitting causes the model to become specific rather than generic. This usually leads to high training accuracy and very low test accuracy.
- Detecting overfitting is useful, but it doesn't solve the actual problem. There are several ways to prevent overfitting, which are stated below:

  - Cross-validation
  - If the training data is too small to train add more relevant and clean data.
  - If the training data is too large, do some feature selection and remove unnecessary features.
  - Regularization

Underfitting

- Underfitting is not often discussed as often as overfitting is discussed. When the model fails to learn from the training dataset and is also not able to generalize the test dataset, is referred to as ***underfitting***. This type of problem can be very easily detected by the performance metrics.
- When a model has high bias and low variance it ends up not generalizing the data and causing underfitting. It is unable to find the hidden underlying patterns from the data. This usually leads to low training accuracy and very low test accuracy. The ways to prevent underfitting are stated below,
- Increase the model complexity
- Increase the number of features in the training data
- Remove noise from the data.

Hands on

- Let's take

Month	Money Spent (X)	Avg(X) – X	[Avg(X) – X]^2	Sales (Y)	Avg(Y) – Y	[Avg(X)-X * Avg(Y)-Y]
1	2000	1300	1,690,000	5000	7600	11,400,000
2	3000	300	90,000	10000	2600	780,000
3	4000	-700	490,000	15000	-2400	1,680,000
4	2500	800	640,000	8000	4600	3,680,000
5	5000	-1700	2,890,000	25000	-12400	21,080,000
AVG(X)	3300			12600		
		SUM XX	5,800,000		SUM YY	37,918,000

Hands on…

- Find the average of the
- X variable, in this case, it is the Money Spent = 3300
- Y variable, in this case, it is Sale = 12600
- Measure the difference between the Average X and individual X
- Square the difference and add the result: **SUM XX = 5, 800,000**
- Multiple the between Avg(X)-X and Avg(Y)-Y and add the results: **SUM XY = 37,918,000**
- To calculate the Slope of the Line, divide the SUM XY by SUM XX
- b (Slope of the line) = 37,918,000 / 5,800,000 = 6.53758621
- To calculate the y-intercept subtract Avg(Y) from Slope * AVG(X)
- a (y-intercept) = 12600 – 6.53758621* 3300 = 8,974.0345
- So, now you have a Simple Linear Regression
- **Y = a + bX**
- **Y = 8,974.0345 + 6.53758621 * X**