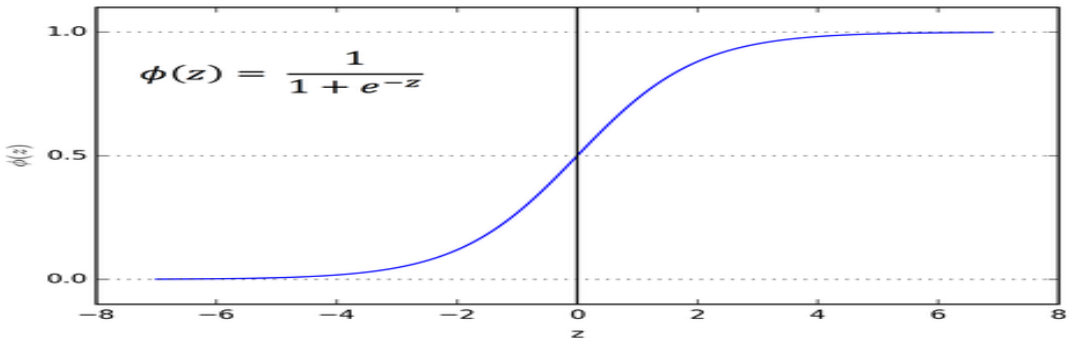
Chapter 4

Machine Learning Algorithms for Classification, Perceptron And Activation Functions

I. Logistic Regression in Machine Learning

- 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 following 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.

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".

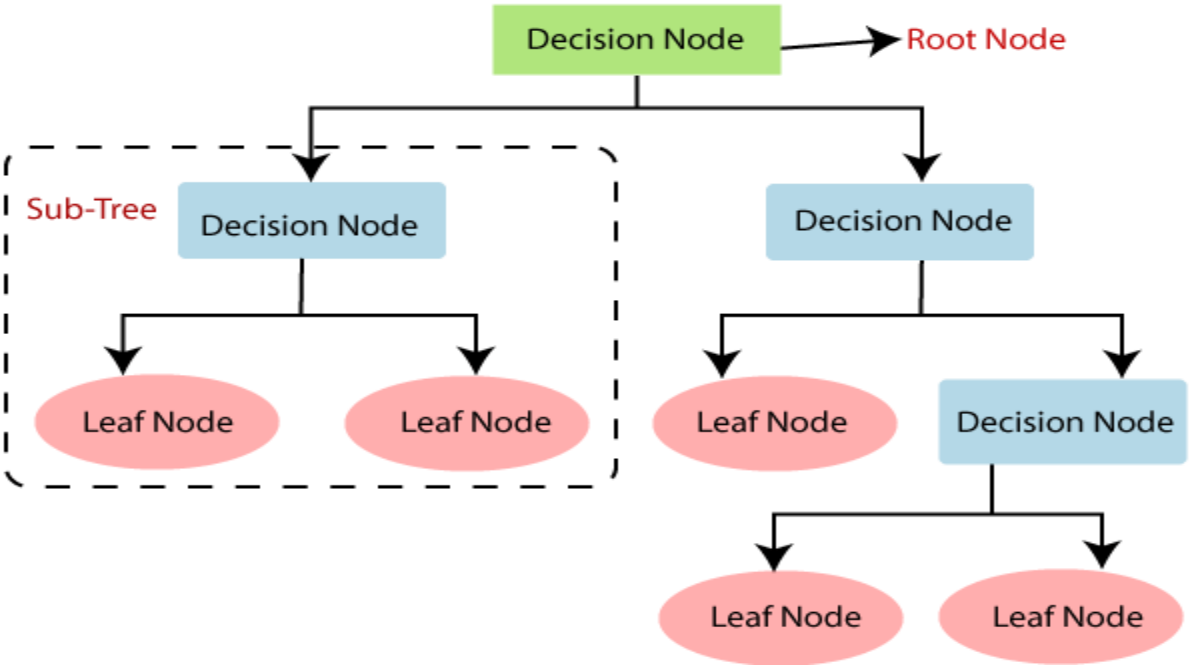
II. Decision Trees

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.
- The following 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 We 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. The following 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:** 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.

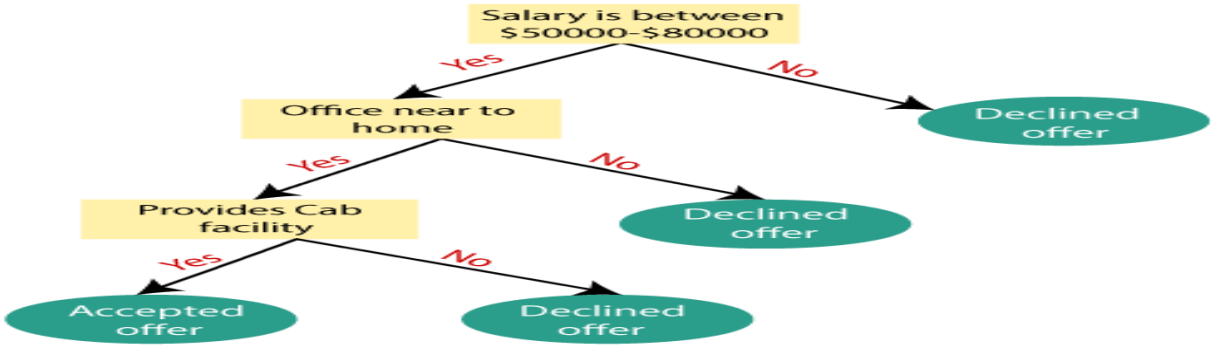
How does the Decision Tree Algorithm Work?

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 following 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.

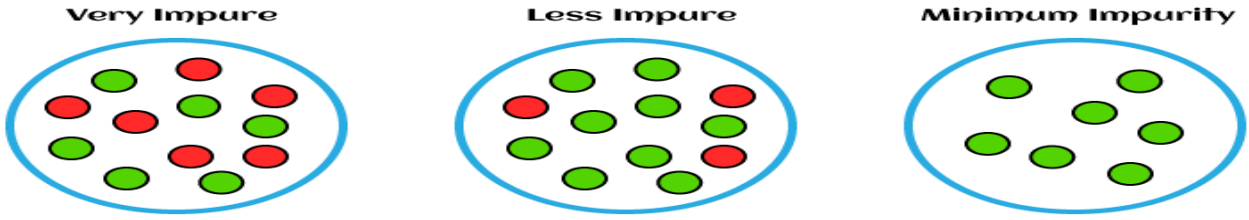
Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) 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 following diagram:



A.Entropy in Machine Learning

We are living in a technology world, and somewhere everything is related to technology. Machine Learning is also the most popular technology in the computer science world that enables the computer to learn automatically from past experiences. Also, Machine Learning is so much demanded in the IT world that most companies want highly skilled machine learning engineers and data scientists for their business. Machine Learning contains lots of algorithms and concepts that solve complex problems easily, and one of them is entropy in Machine Learning. Almost everyone must have heard the Entropy word once during their school or college days in physics and chemistry. The base of entropy comes from physics, where it is defined as the measurement of disorder, randomness, unpredictability, or impurity in the system.

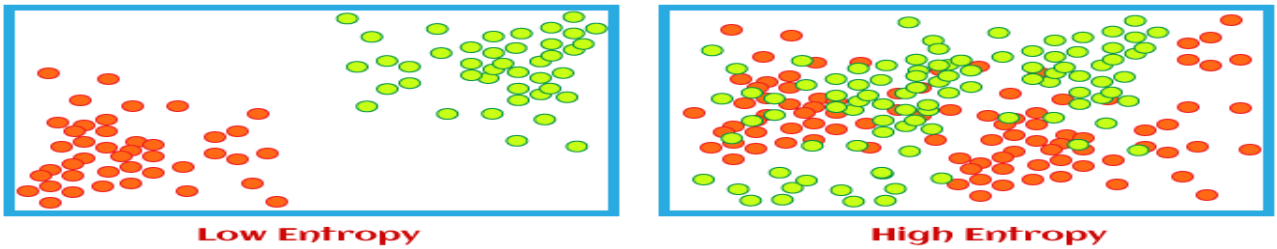
From Machine Learning side, Entropy is defined as the randomness or measuring the disorder of the information being processed in Machine Learning. Further, in other words, we can say that entropy is the machine learning metric that measures the unpredictability or impurity in the system**.**



When information is processed in the system, then every piece of information has a specific value to make and can be used to draw conclusions from it. So, if it is easier to draw a valuable conclusion from a piece of information, then entropy will be lower in Machine learning, or if entropy is higher, then it will be difficult to draw any conclusion from that piece of information.
Entropy is a useful tool in machine learning to understand various concepts such as feature selection, building decision trees, and fitting classification models, etc. Being a machine learning engineer and professional data scientist, you must have in-depth knowledge of entropy in machine learning.

What is Entropy in Machine Learning

Entropy is the measurement of disorder or impurities in the information processed in machine learning. It determines how a decision tree chooses to split data.



We can understand the term entropy with any simple example: flipping a coin. When we flip a coin, then there can be two outcomes. However, it is difficult to conclude what would be the exact outcome while flipping a coin because there is no direct relation between flipping a coin and its outcomes. There is a 50% probability of both outcomes; then, in such scenarios, entropy would be high. This is the essence of entropy in machine learning.

Mathematical Formula for Entropy

Consider a data set having a total number of N classes, then the entropy (E) can be determined with the formula:

$$E = -\sum_{i=1}^{N} P_i \log_2 P_i$$

Where;
P_i = Probability of randomly selecting an example in class I;
Entropy always lies between 0 and 1, however depending on the number of classes in the dataset, it can be greater than 1.
Let's understand it with an example where we have a dataset having three colors of fruits as red, green, and yellow. Suppose we have 2 red, 2 green, and 4 yellow observations throughout the dataset. Then as per the above equation:
E=−(prlog2pr+pglog2pg+pylog2py)
Where;
P_r = Probability of choosing red fruits;
P_g = Probability of choosing green fruits and;
P_y = Probability of choosing yellow fruits.
P_r = 2/8 =1/4 [As only 2 out of 8 datasets represents red fruits]
P_g = 2/8 =1/4 [As only 2 out of 8 datasets represents green fruits]
P_y = 4/8 = 1/2 [As only 4 out of 8 datasets represents yellow fruits]
Now our final equation will be such as;

$$E = -\left(\frac{1}{4}\log_2\left(\frac{1}{4}\right) + \frac{1}{4}\log_2\left(\frac{1}{4}\right) + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right)$$

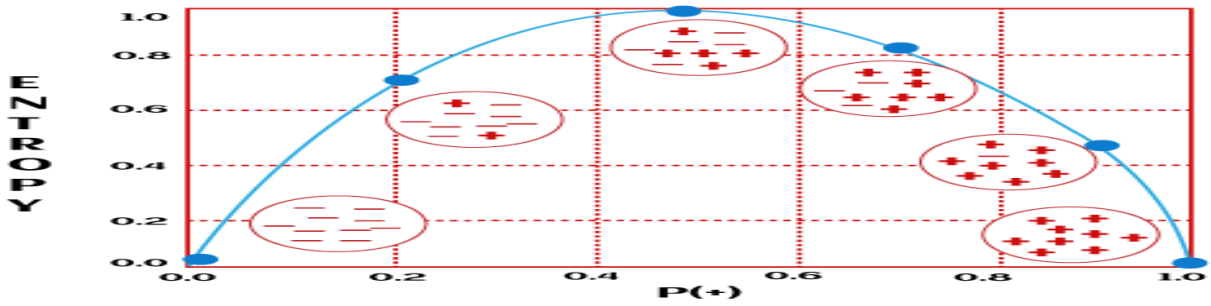
$$E = -(1/4*(-2)+1/4(-2)+1/2*(-1)$$

$$E = -(-1/2-1/2-1/2)$$

$$E = -(-1.5) = 1.5$$

So, entropy will be **1.5.**

Let's consider a case when all observations belong to the same class; then entropy will always be 0.
E=−(1log21) = 0
When entropy becomes 0, then the dataset has no impurity. Datasets with 0 impurities are not useful for learning. Further, if the entropy is 1, then this kind of dataset is good for learning.



B. Attribute Selection Measures (ASM)

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 (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**

i. Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation 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 following 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:

$$\mathbf{Entropy(S) = -P(yes)\log_2 P(yes) - P(no)\log_2 P(no)}$$

Where,

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

ii. 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 following formula:

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

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.

III. Bayes Theorem in Machine learning

Machine Learning is one of the most emerging technologies of Artificial Intelligence. We are living in the 21th century which is completely driven by new technologies and gadgets in which some are yet to be used and few are on its full potential. Similarly, Machine Learning is also a technology that is still in its developing phase. There are lots of concepts that make machine learning a better technology such as supervised learning, unsupervised learning, reinforcement learning, perceptron models, Neural networks, etc.

"Bayes Theorem in Machine Learning", is another most important concept of Machine Learning theorem i.e., ***Bayes Theorem***. But before starting this topic you should have essential understanding of this theorem such as what exactly is Bayes theorem, why it is used in Machine Learning, examples of Bayes theorem in Machine Learning and much more.

Bayes theorem is given by an English statistician, philosopher, and Presbyterian minister named **Mr. Thomas Bayes** in 17^{th} century. Bayes provides their thoughts in decision theory which is extensively used in important mathematics concepts such as Probability. Bayes theorem is also widely used in Machine Learning where we need to predict classes precisely and accurately. An important concept of Bayes theorem named **Bayesian method** is used to calculate conditional probability in Machine Learning application that includes classification tasks. Further, a simplified version of Bayes theorem (Naïve Bayes classification) is also used to reduce computation time and average cost of the projects.

Bayes theorem is also known with some other name such as **Bayes rule or Bayes Law.** ***Bayes*** theorem helps to determine the probability of an event with random knowledge. It is used to calculate the probability of occurring one event while other one already occurred. It is a best method to relate the condition probability and marginal probability.

In simple words, we can say that Bayes theorem helps to contribute more accurate results.

Bayes Theorem is used to estimate the precision of values and provides a method for calculating the conditional probability. However, it is hypocritically a simple calculation but it is used to easily calculate the conditional probability of events where intuition often fails. Some of the data scientist assumes that

Bayes theorem is most widely used in financial industries but it is not like that. Other than financial, Bayes theorem is also extensively applied in health and medical, research and survey industry, aeronautical sector, etc.

What is Bayes Theorem?

Bayes theorem is one of the most popular machine learning concepts that helps to calculate the probability of occurring one event with uncertain knowledge while other one has already occurred. Bayes' theorem can be derived using product rule and conditional probability of event X with known event Y:

- According to the product rule we can express as the probability of event X with known event Y as follows;

1. P(X ? Y)= P(X|Y) P(Y) {equation 1}

- Further, the probability of event Y with known event X:

1. P(X ? Y)= P(Y|X) P(X) {equation 2}

Mathematically, Bayes theorem can be expressed by combining both equations on right hand side. We will get:

$$P(X|Y) = \frac{P(Y|X).P(X)}{P(Y)}$$

Here, both events X and Y are independent events which means probability of outcome of both events does not depends one another.
The above equation is called as Bayes Rule or Bayes Theorem.

- P(X|Y) is called as **posterior**, which we need to calculate. It is defined as updated probability after considering the evidence.
- P(Y|X) is called the **likelihood**. It is the probability of evidence when hypothesis is true.
- P(X) is called the **prior probability**, probability of hypothesis before considering the evidence
- P(Y) is called **marginal probability**. It is defined as the probability of evidence under any consideration.

Hence, Bayes Theorem can be written as: **posterior = likelihood * prior / evidence**

Prerequisites for Bayes Theorem

While studying the Bayes Theorem, we need to understand few important concepts. These are:

1. Experiment: An experiment is defined as the planned operation carried out under controlled condition such as tossing a coin, drawing a card and rolling a dice, etc.
2. Sample Space: During an experiment what we get as a result is called as possible outcomes and the set of all possible outcome of an event is known as sample space. For example, if we are rolling a dice, sample space will be: S1 = {1, 2, 3, 4, 5, 6}. Similarly, if our experiment is related to toss a coin and recording its outcomes, then sample space will be: S2 = {Head, Tail}
3. Event: Event is defined as subset of sample space in an experiment. Further, it is also called as set of outcomes.

Assume in our experiment of rolling a dice, there are two event A and B such that;
A = Event when an even number is obtained = {2, 4, 6}
B = Event when a number is greater than 4 = {5, 6}

- **Probability of the event A "P(A)"**= Number of favourable outcomes / Total number of possible outcomes: P(E) = 3/6 =1/2 =0.5
- Similarly, **Probability of the event B "P(B)"**= Number of favourable outcomes / Total number of possible outcomes =2/6 =1/3 =0.333
- **Union of event A and B:** A∪B = {2, 4, 5, 6}
- **Intersection of event A and B:** A∩B= {6}
- **Disjoint Event:** If the intersection of the event A and B is an empty set or null then such events are known as **disjoint event** or **mutually exclusive events** also.

4. Random Variable: It is a real value function which helps mapping between sample space and a real line of an experiment. A random variable is taken on some random values and each value having some probability. However, it is neither random nor a variable but it behaves as a function which can either be discrete, continuous or combination of both.

5. Exhaustive Event: As per the name suggests, a set of events where at least one event occurs at a time, called exhaustive event of an experiment. Thus, two events A and B are said to be exhaustive if either A or B definitely occur at a time and both are mutually exclusive for e.g., while tossing a coin, either it will be a Head or may be a Tail.

6. Independent Event: Two events are said to be independent when occurrence of one event does not affect the occurrence of another event. In simple words we can say that the probability of outcome of both events does not depends one another. Mathematically, two events A and B are said to be independent if: P(A ∩ B) = P(AB) = P(A)*P(B)

7. Conditional Probability: Conditional probability is defined as the probability of an event A, given that another event B has already occurred (i.e. A conditional B). This is represented by P(A|B) and we can define it as: P(A|B) = P(A ∩ B) / P(B)

8. Marginal Probability: Marginal probability is defined as the probability of an event A occurring independent of any other event B. Further, it is considered as the probability of evidence under any consideration. P(A) = P(A|B)*P(B) + P(A|~B)*P(~B)

Here ~B represents the event that B does not occur.

How to apply Bayes Theorem or Bayes rule in Machine Learning?

Bayes theorem helps us to calculate the single term P(B|A) in terms of P(A|B), P(B), and P(A). This rule is very helpful in such scenarios where we have a good probability of P(A|B), P(B), and P(A) and need to determine the fourth term.

Naïve Bayes classifier is one of the simplest applications of Bayes theorem which is used in classification algorithms to isolate data as per accuracy, speed and classes. Let's understand the use of Bayes theorem in machine learning with the following example.

Suppose, we have a vector A with I attributes. It means: A = A1, A2, A3, A4……Ai and Further, we have n classes represented as C1, C2, C3, C4…….Cn. These are two conditions given to us, and our classifier that works on Machine Language has to predict A and the first thing that our classifier has to choose will be the best possible class. So, with the help of Bayes theorem, we can write it as:

P(Ci/A)= [P(A/Ci) * P(Ci)] / P(A), Here; P(A) is the condition-independent entity.

P(A) will remain constant throughout the class means it does not change its value with respect to change in class. To maximize the P(Ci/A), we have to maximize the value of term P(A/Ci) * P(Ci). With n number classes on the probability list let's assume that the possibility of any class being the right answer is equally likely. Considering this factor, we can say that:

P(C1)=P(C2)-P(C3)=P(C4)=…..=P(Cn).

This process helps us to reduce the computation cost as well as time. This is how Bayes theorem plays a significant role in Machine Learning and Naïve Bayes theorem has simplified the conditional probability tasks without affecting the precision. Hence, we can conclude that:

P(Ai/C)= P(A1/C)* P(A2/C)* P(A3/C)*……*P(An/C)

Hence, by using Bayes theorem in Machine Learning we can easily describe the possibilities of smaller events.

What is Naïve Bayes Classifier in Machine Learning

Naïve Bayes theorem is also a supervised algorithm, which is based on Bayes theorem and used to solve classification problems. It is one of the most simple and effective classification algorithms in Machine Learning which enables us to build various ML models for quick predictions. It is a probabilistic classifier that means it predicts on the basis of probability of an object. Some popular Naïve Bayes algorithms are **spam filtration, Sentimental analysis, and classifying articles.**

Advantages of Naïve Bayes Classifier in Machine Learning:

- It is one of the simplest and effective methods for calculating the conditional probability and text classification problems.
- A Naïve-Bayes classifier algorithm is better than all other models where assumption of independent predictors holds true.
- It is easy to implement than other models.
- It requires small amount of training data to estimate the test data which minimize the training time period.
- It can be used for Binary as well as Multi-class Classifications.

Disadvantages of Naïve Bayes Classifier in Machine Learning:

The main disadvantage of using Naïve Bayes classifier algorithms is, it limits the assumption of independent predictors because it implicitly assumes that all attributes are independent or unrelated but in real life it is not feasible to get mutually independent attributes.

Conclusion

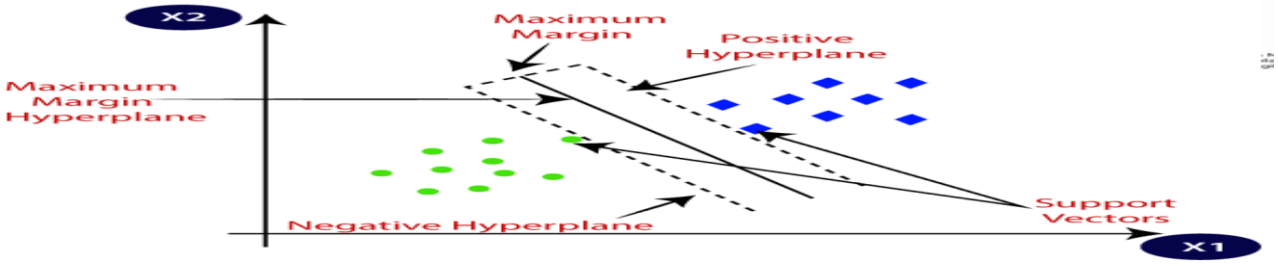
Though, we are living in technology world where everything is based on various new technologies that are in developing phase but still these are incomplete in absence of already available classical theorems and algorithms. Bayes theorem is also the most popular example that is used in Machine Learning. Bayes theorem has so many applications in Machine Learning. In classification related problems, it is one of the

most preferred methods than all other algorithms. Hence, we can say that Machine Learning is highly dependent on Bayes theorem.

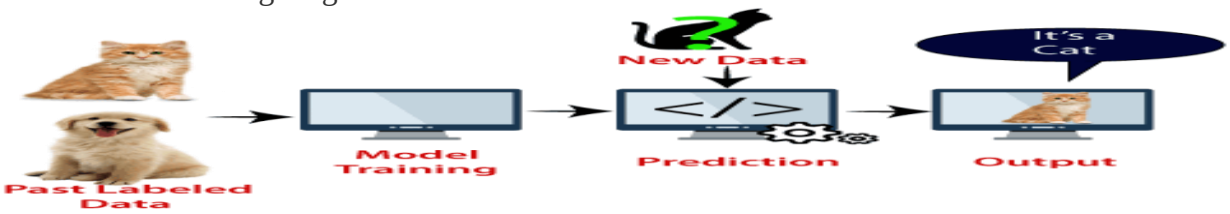
IV. Support Vector Machine

Support Vector Machine (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 the algorithm is termed as Support Vector Machine. Consider the following 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 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 following 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.

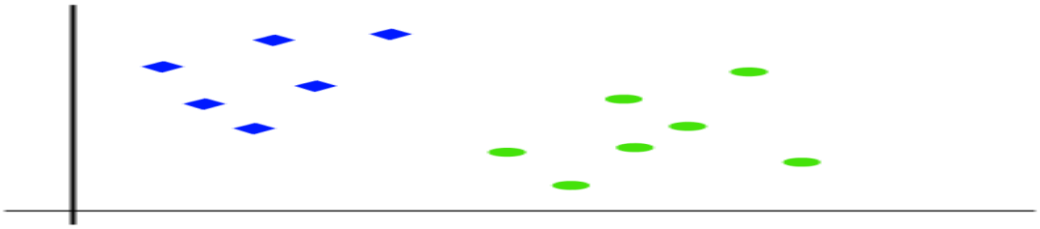
Hyperplane and Support Vectors in the SVM algorithm:

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.

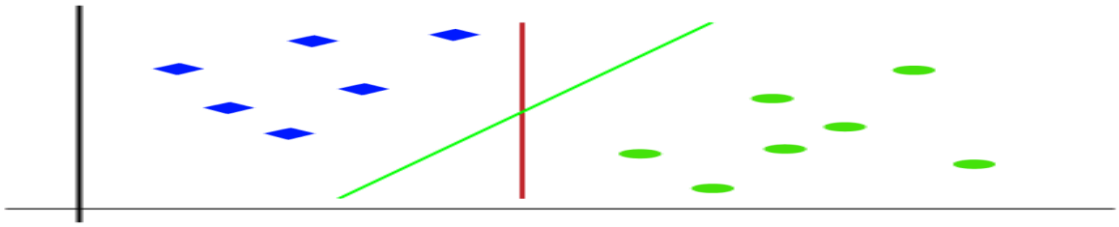
Support Vectors: The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vectors. Since these vectors support the hyperplane, hence called a Support vector.

How does SVM works?

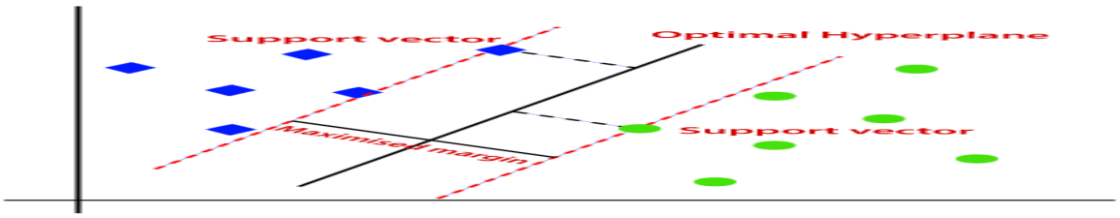
Linear SVM: The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the following image:



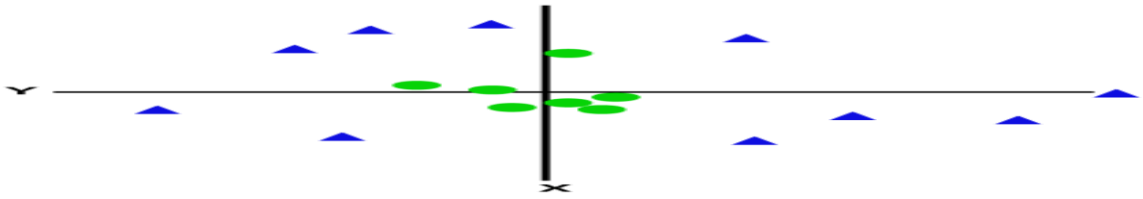
So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the following image:



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**.

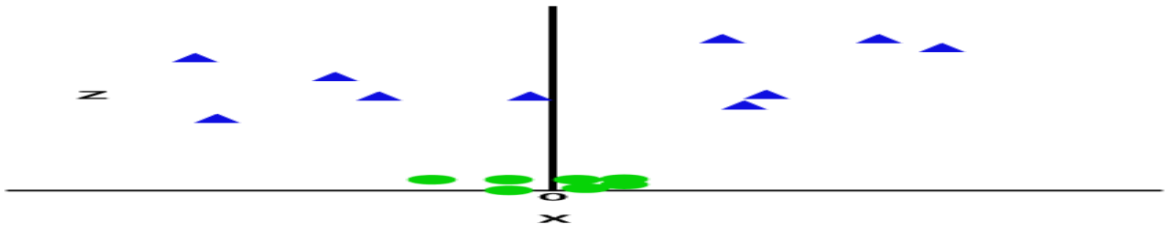


Non-Linear SVM: If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the following image:

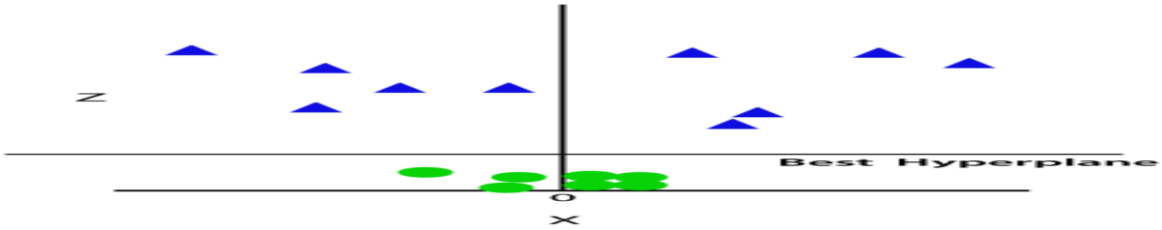


So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third dimension z. It can be calculated as: **$z=x^2+y^2$**

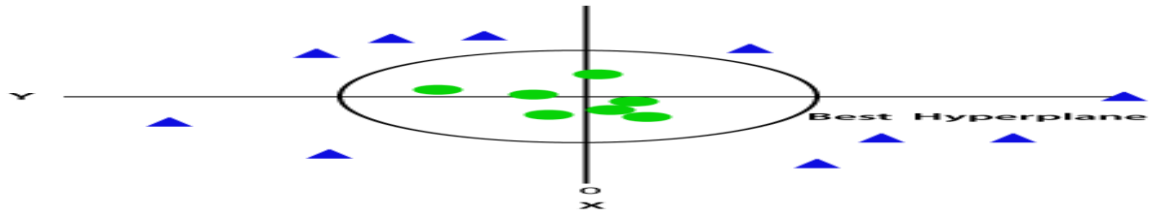
By adding the third dimension, the sample space will become as the following image:



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

V. Perceptron in Machine Learning

In Machine Learning and Artificial Intelligence, Perceptron is the most commonly used term for all folks. It is the primary step to learn Machine Learning and Deep Learning technologies, which consists of a set of weights, input values or scores, and a threshold. Perceptron is a building block of an Artificial Neural Network. Initially, in the mid of 19th century, **Mr. Frank Rosenblatt** invented the Perceptron for performing certain calculations to detect input data capabilities or business intelligence. Perceptron is a

linear Machine Learning algorithm used for supervised learning for various binary classifiers. This algorithm enables neurons to learn elements and processes them one by one during preparation

So, what is the Perceptron model in Machine Learning?

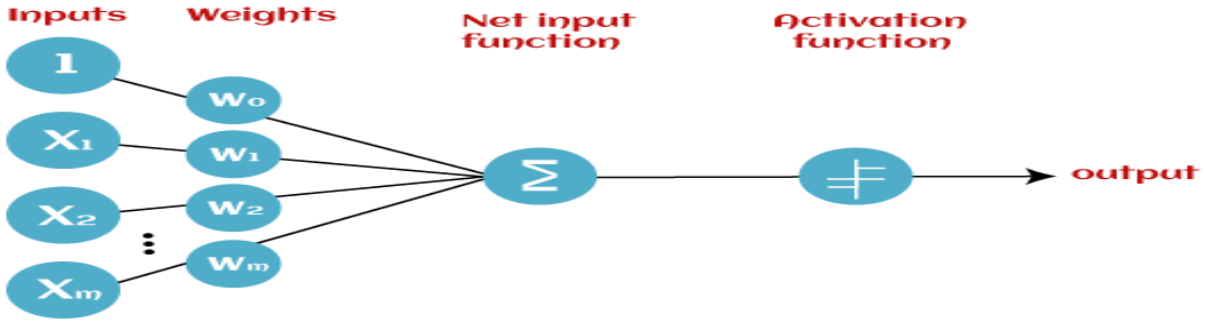
Perceptron is a Machine Learning algorithm for supervised learning of various binary classification tasks. Further, ***Perceptron is also understood as an Artificial Neuron or neural network unit that helps to detect certain input data computations in business intelligence***.

Perceptron model is also treated as one of the best and simplest types of **Artificial Neural Networks (ANNs)**. However, it is a supervised learning algorithm of binary classifiers. Hence, we can consider it as a single-layer neural network with four main parameters, i.e., input values, weights and Bias, net sum, and an activation function.

What is Binary classifier in Machine Learning?

In Machine Learning, binary classifiers are defined as the function that helps in deciding whether input data can be represented as vectors of numbers and belongs to some specific class. Binary classifiers can be considered as linear classifiers. In simple words, we can understand it as *a* classification algorithm that can predict linear predictor function in terms of weight and feature vectors.

Basic Components of Perceptron: Mr. Frank Rosenblatt invented the perceptron model as a binary classifier which contains three main components. These are as follows:

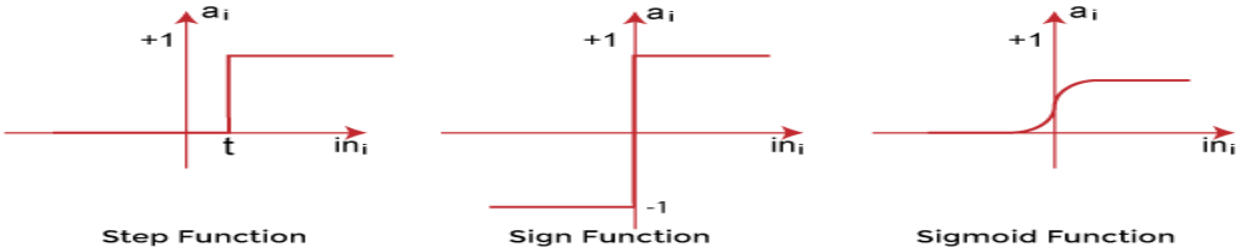


Input Nodes or Input Layer: is the primary component of Perceptron which accepts the initial data into the system for further processing. Each input node contains a real numerical value.

Weight and Bias: Weight parameter represents the strength of the connection between units. This is another most important parameter of Perceptron components. Weight is directly proportional to the strength of the associated input neuron in deciding the output. Further, Bias can be considered as the line of intercept in a linear equation.

Activation Function: These are the final and important components that help to determine whether the neuron will fire or not. Activation Function can be considered primarily as a step function. Types of Activation functions:

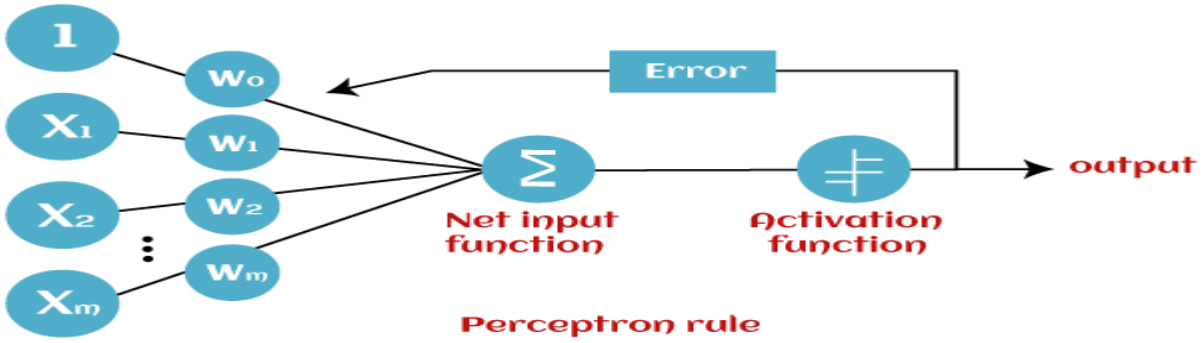
- **Sign function**
- **Step function, and**
- **Sigmoid function**



The data scientist uses the activation function to take a subjective decision based on various problem statements and forms the desired outputs. Activation function may differ (e.g., Sign, Step, and Sigmoid) in perceptron models by checking whether the learning process is slow or has vanishing or exploding gradients.

How does Perceptron Work?

In Machine Learning, Perceptron is considered as a single-layer neural network that consists of four main parameters named input values (Input nodes), weights and Bias, net sum, and an activation function. The perceptron model begins with the multiplication of all input values and their weights, then adds these values together to create the weighted sum. Then this weighted sum is applied to the activation function 'f' to obtain the desired output. This activation function is also known as the **step function** and is represented by '**f**'.



This step function or Activation function plays a vital role in ensuring that output is mapped between required values (0,1) or (-1,1). It is important to note that the weight of input is indicative of the strength of a node. Similarly, an input's bias value gives the ability to shift the activation function curve up or down.

Perceptron model works in two important steps as follows:

Step-1: In the first step first, multiply all input values with corresponding weight values and then add them to determine the weighted sum. Mathematically, we can calculate the weighted sum as follows: **∑wi*xi = x1*w1 + x2*w2 +…wn*xn,** then add a special term called **bias 'b'** to this weighted sum to improve the model's performance.

∑wi*xi + b

Step-2: In the second step, an activation function is applied with the above-mentioned weighted sum, which gives us output either in binary form or a continuous value as follows:

Y = f(∑wi*xi + b)

Types of Perceptron Models

Based on the layers, Perceptron models are divided into two types. These are as follows:

1. Single-layer Perceptron Model
2. Multi-layer Perceptron model

Single Layer Perceptron Model: This is one of the easiest Artificial neural networks (ANN) types. A single-layered perceptron model consists feed-forward network and also includes a threshold transfer function inside the model. The main objective of the single-layer perceptron model is to analyze the linearly separable objects with binary outcomes.

In a single layer perceptron model, its algorithms do not contain recorded data, so it begins with inconstantly allocated input for weight parameters. Further, it sums up all inputs (weight). After adding all inputs, if the total sum of all inputs is more than a pre-determined value, the model gets activated and shows the output value as +1.

If the outcome is same as pre-determined or threshold value, then the performance of this model is stated as satisfied, and weight demand does not change. However, this model consists of a few discrepancies triggered when multiple weight inputs values are fed into the model. Hence, to find desired output and minimize errors, some changes should be necessary for the weights input.

"Single-layer perceptron can learn only linearly separable patterns."

Multi-Layered Perceptron Model: will be discussed in next chapter.

Perceptron Function

Perceptron function "f(x)" can be achieved as output by multiplying the input 'x' with the learned weight coefficient 'w'. Mathematically, we can express it as follows:

f(x)=1; if w.x+b>0

otherwise, f(x)=0

- 'w' represents real-valued weights vector
- 'b' represents the bias, 'x' represents a vector of input x values.

Characteristics of Perceptron

The perceptron model has the following characteristics.

1. Perceptron is a machine learning algorithm for supervised learning of binary classifiers.
2. In Perceptron, the weight coefficient is automatically learned.
3. Initially, weights are multiplied with input features, and the decision is made whether the neuron is fired or not.
4. The activation function applies a step rule to check whether the weight function is greater than zero.
5. The linear decision boundary is drawn, enabling the distinction between the two linearly separable classes +1 and -1.
6. If the added sum of all input values is more than the threshold value, it must have an output signal; otherwise, no output will be shown.

Limitations of Perceptron Model

A perceptron model has limitations:

- The output of a perceptron can only be a binary number (0 or 1) due to the hard limit transfer function.
- Perceptron can only be used to classify the linearly separable sets of input vectors. If input vectors are non-linear, it is not easy to classify them properly.

Future of Perceptron

The future of the Perceptron model is much bright and significant as it helps to interpret data by building intuitive patterns and applying them in the future. Machine learning is a rapidly growing technology of Artificial Intelligence that is continuously evolving and in the developing phase; hence the future of perceptron technology will continue to support and facilitate analytical behavior in machines that will, in turn, add to the efficiency of computers. The perceptron model is continuously becoming more advanced and working efficiently on complex problems with the help of artificial neurons.

Conclusion: you have learned how Perceptron models are the simplest type of artificial neural network which carries input and their weights, the sum of all weighted input, and an activation function. Perceptron models are continuously contributing to Artificial Intelligence and Machine Learning, and these models are becoming more advanced. Perceptron enables the computer to work more efficiently on complex problems using various Machine Learning technologies. The Perceptrons are the fundamentals of artificial neural networks, and everyone should have in-depth knowledge of perceptron models to study deep neural networks.

VI. Activation Functions in Neural Networks

What is Activation Function? It's just a thing function that you use to get the output of node. It is also known as **Transfer Function**.

Why we use Activation functions with Neural Networks?

It is used to determine the output of neural network like yes or no. It maps the resulting values in between 0 to 1 or -1 to 1 etc. (depending upon the function).

The Activation Functions can be basically divided into 2 types.

1. Linear Activation Function
2. Non-linear Activation Functions

Linear or Identity Activation Function

As you can see the function is a line or linear. Therefore, the output of the functions will not be confined between any range.

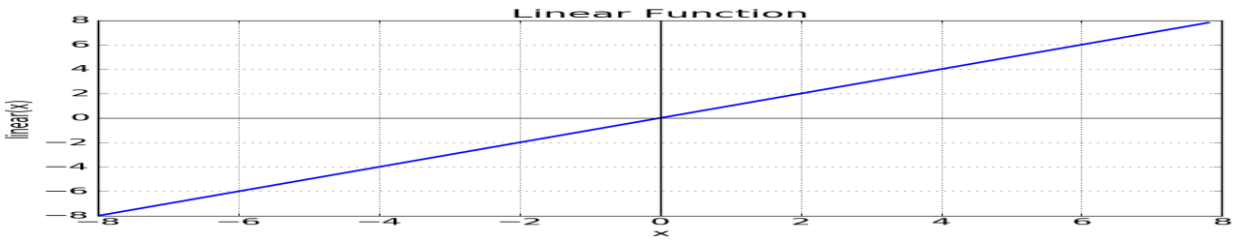


Fig: Linear Activation Function

Equation : f(x) = x And **Range :** (-infinity to infinity)

It doesn't help with the complexity or various parameters of usual data that is fed to the neural networks.

Non-linear Activation Function

The Nonlinear Activation Functions are the most widely used activation functions. Nonlinearity helps to make the graph look something like this

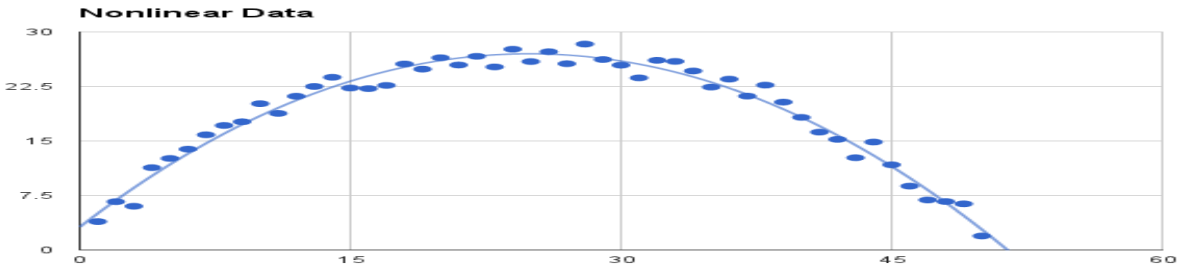


Fig: Non-linear Activation Function

It makes it easy for the model to generalize or adapt with variety of data and to differentiate between the output. The main terminologies needed to understand for nonlinear functions are:
Derivative or Differential: Change in y-axis w.r.t. change in x-axis.It is also known as slope.
Monotonic function*:* A function which is either entirely non-increasing or non-decreasing.
The Nonlinear Activation Functions are mainly divided on the basis of their **range or curves**.

1. **Sigmoid** or **Logistic Activation Function**

The Sigmoid Function curve looks like a S-shape.

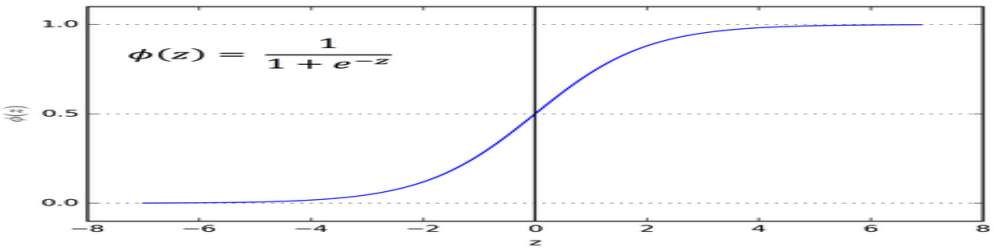


Fig-1: Sigmoid Function

The main reason why we use sigmoid function is because it exists between **(0 to 1).** Therefore, it is especially used for models where we have to **predict the probability** as an output. Since probability of anything exists only between the range of **0 and 1,** sigmoid is the right choice.
The function is differentiable. That means, we can find the slope of the sigmoid curve at any two points.
The function is **monotonic** but function's derivative is not.
The logistic sigmoid function can cause a neural network to get stuck at the training time.
The **softmax function** is a more generalized logistic activation function which is used for multiclass classification.

2. **Tanh or hyperbolic tangent Activation Function**

tanh is also like logistic sigmoid but better. The range of the tanh function is from (-1 to 1). tanh is also sigmoidal (s - shaped).

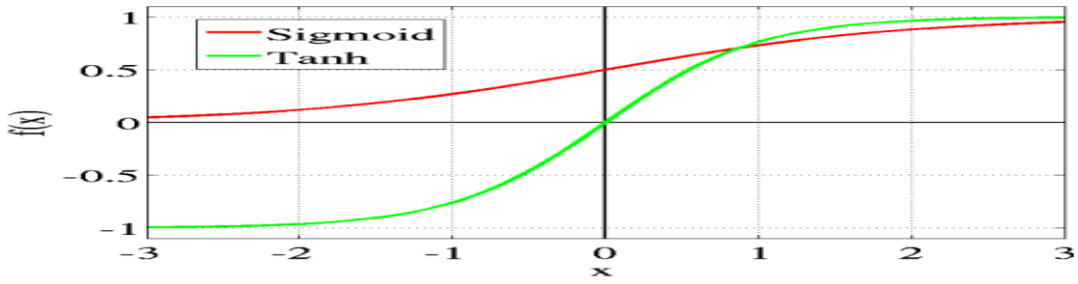


Fig-2: tanh v/s Logistic Sigmoid

The advantage is that the negative inputs will be mapped strongly negative and the zero inputs will be mapped near zero in the tanh graph.
The function is differentiable**.**
The function is **monotonic** while its derivative is not monotonic**.**
The tanh function is mainly used classification between two classes.
Both tanh and logistic sigmoid activation functions are used in feed-forward nets.

3. **ReLU (Rectified Linear Unit) Activation Function**

The ReLU is the most widely used activation function in the world right now since, it is used in almost all the convolutional neural networks or deep learning.

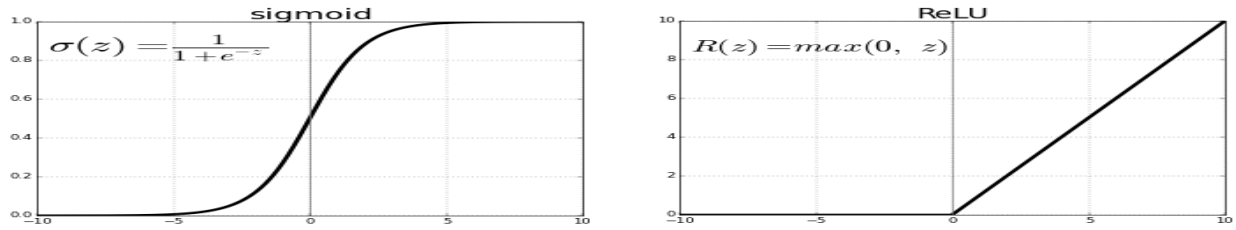


Fig-3: ReLU v/s Logistic Sigmoid

As you can see, the ReLU is half rectified (from bottom). f(z) is zero when z is less than zero and f(z) is equal to z when z is above or equal to zero.
Range: [0 to infinity)
The function and its derivative both are monotonic**.**
But the issue is that all the negative values become zero immediately which decreases the ability of the model to fit or train from the data properly. That means any negative input given to the ReLU activation function turns the value into zero immediately in the graph, which in turns affects the resulting graph by not mapping the negative values appropriately.

4. Leaky ReLU

It is an attempt to solve the dying ReLU problem

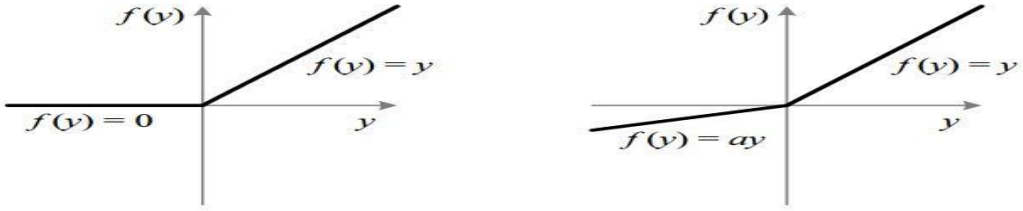


Fig-4: ReLU v/s Leaky ReLU

The leak helps to increase the range of the ReLU function. Usually, the value of **a** is 0.01 or so.

When **a is not 0.01** then it is called **Randomized ReLU**.
Therefore the **range** of the Leaky ReLU is (-infinity to infinity).
Both Leaky and Randomized ReLU functions are monotonic in nature. Also, their derivatives is monotonic in nature.