Unit 3 : Supervised Learning - II

Dr. Vinod Patidar
Associate Professor
Computer Science and Engineering

Content

- K-NN classifier
- Logistic regression
- Perceptron
- Multi-layer Perceptron
- Linear SVM
- Non-linear SVM

K-Nearest Neighbor(KNN) Algorithm

- K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.

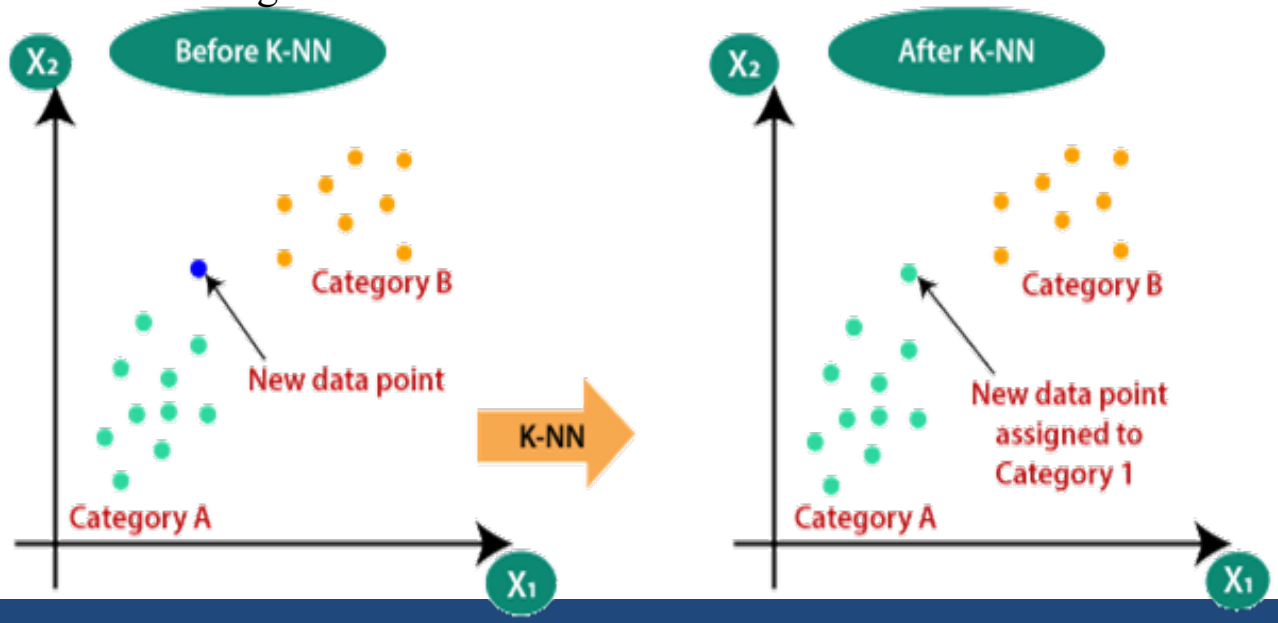
K-Nearest Neighbor(KNN) Algorithm

- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
- Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

K-Nearest Neighbor(KNN) Algorithm

Why do we need a K-NN Algorithm?

- Suppose there are two categories, i.e. Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.
- Consider the below diagram:.

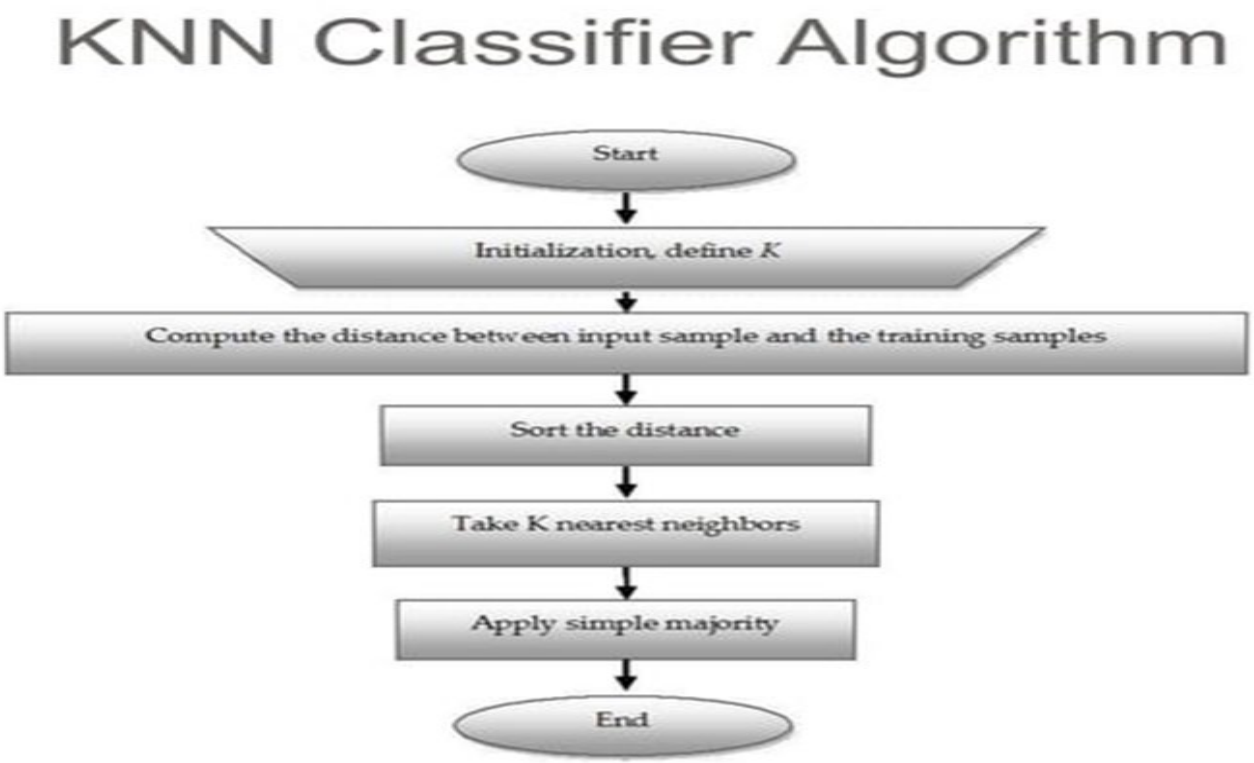


How does K-NN work?

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

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

KNN Algorithm

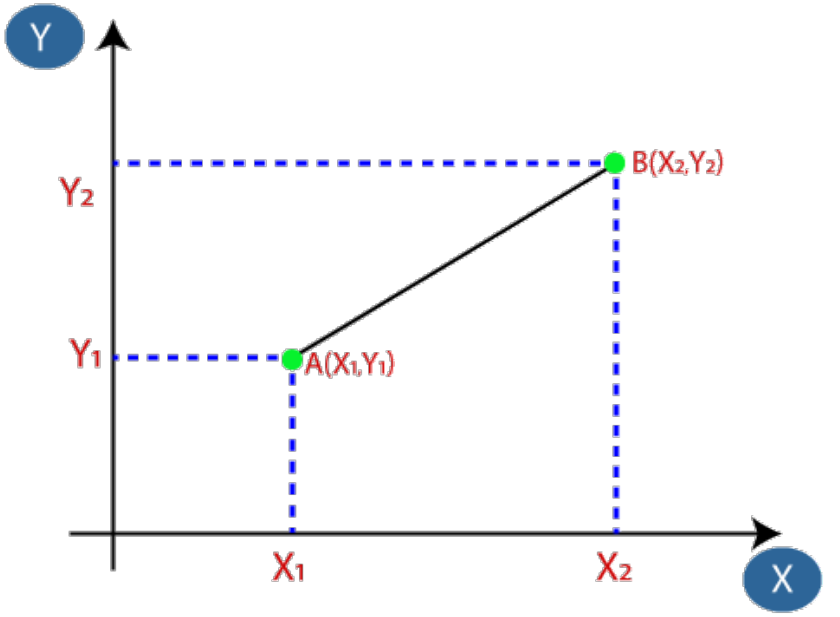


How does K-NN work?

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

How does K-NN work?

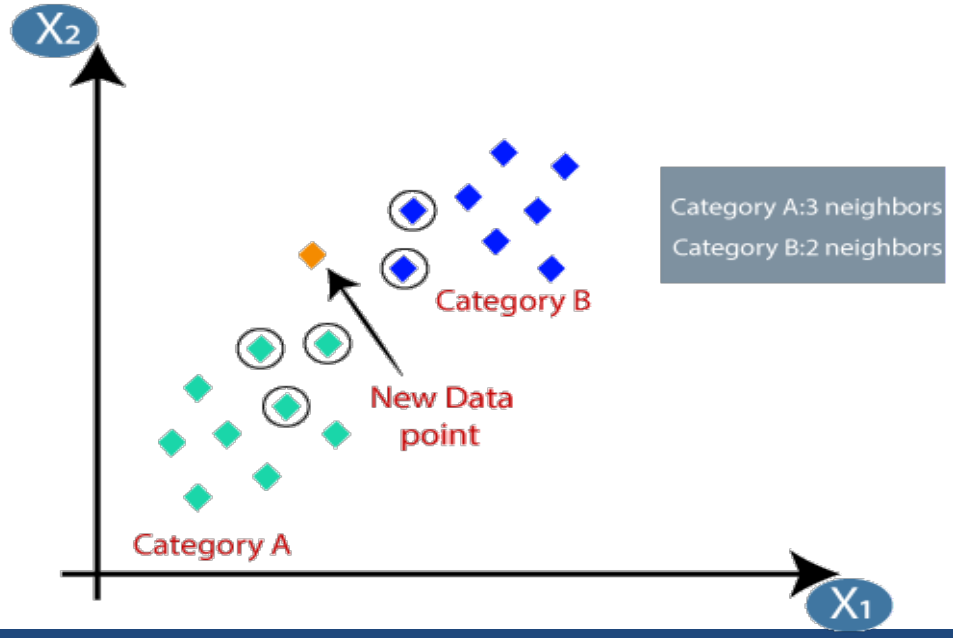
- Firstly, we will choose the number of neighbors, so we will choose the k=5.
- Next, we will calculate the Euclidean distance between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:



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

How does K-NN work?

- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the Given image:
- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.



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

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

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

Advantages and Disadvantages of KNN Algorithm:

Advantages

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

Disadvantages

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

KNN Algorithm Example

Distance Measures

$$Euclidean\ distance: d(x,y) = \sqrt{\sum (x_i - yi)^2}$$

$$Squared\ Euclidean\ distance: d(x,y) = \sum (x_i - yi)^2$$

$$Manhattan\ distance: d(x,y) = \sum |(xi - yi)|$$

Which distance measure to use?

We use Euclidean Distance as it treats each feature as equally important.

KNN Algorithm Example

Example

- We have data from the questionnaires survey and objective testing with two attributes (acid durability and strength) to classify whether a special paper tissue is good or not. Here are four training samples :

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Y = Classification
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Now the factory produces a new paper tissue that passes the laboratory test with X1 = 3 and X2 = 7. Guess the classification of this new tissue.

KNN Algorithm Example

- Step 1 : **Initialize and Define k.**

  Lets say, k = 3

  (Always choose k as an odd number if the number of attributes is even to avoid a tie in the class prediction)

- Step 2 : **Compute the distance between input sample and training sample**

  - Co-ordinate of the input sample is (3,7).

  - Instead of calculating the Euclidean distance, we calculate the Squared Euclidean distance.

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Squared Euclidean distance
7	7	$(7-3)^2 + (7-7)^2 = \mathbf{16}$
7	4	$(7-3)^2 + (4-7)^2 = \mathbf{25}$
3	4	$(3-3)^2 + (4-7)^2 = \mathbf{09}$
1	4	$(1-3)^2 + (4-7)^2 = \mathbf{13}$

KNN Algorithm Example

- Step 3 : **Sort the distance and determine the nearest neighbours based of the Kth minimum distance :**

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Squared Euclidean distance	Rank minimum distance	Is it included in 3-Nearest Neighbour?
7	7	16	3	Yes
7	4	25	4	No
3	4	09	1	Yes
1	4	13	2	Yes

KNN Algorithm Example

- Step 4 : **Take 3-Nearest Neighbours:**
- Gather the category Y of the nearest neighbours.

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Squared Euclidean distance	Rank minimum distance	Is it included in 3-Nearest Neighbour?	Y = Category of the nearest neighbour
7	7	16	3	Yes	Bad
7	4	25	4	No	-
3	4	09	1	Yes	Good
1	4	13	2	Yes	Good

KNN Algorithm Example

- Step 5 : **Apply simple majority**
- Use simple majority of the category of the nearest neighbours as the prediction value of the query instance.
- We have 2 "good" and 1 "bad". Thus we conclude that the new paper tissue that passes the laboratory test with X1 = 3 and X2 = 7 is included in the "good" category.

Applications of KNN Classifier

- Used in classification
- Used to get missing values
- Used in pattern recognition
- Used in gene expression
- Used in protein-protein prediction
- Used to get 3D structure of protein
- Used to measure document similarity

Logistic Regression

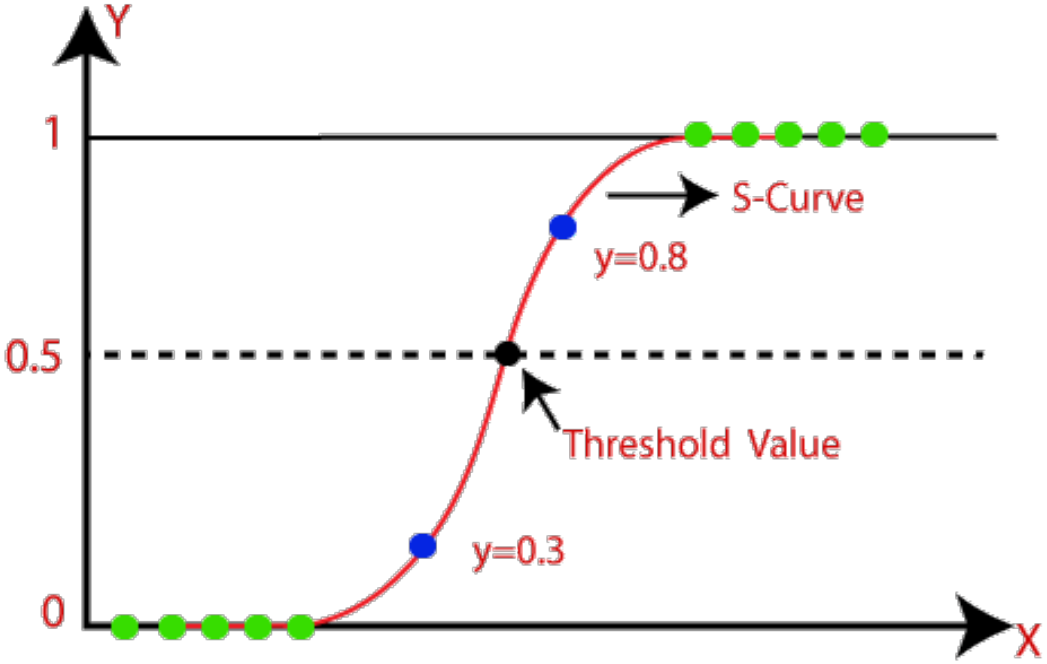
- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

Logistic Regression

- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- It is based on the concept of regression, but instead of predicting continuous values, it predicts the probability of an instance belonging to a particular class.
- Logistic Regression is widely used in various domains, including healthcare, finance, marketing, and social sciences.
- It is especially useful when the dependent variable (the variable being predicted) is categorical or binary

Logistic Regression

- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Logistic Function (Sigmoid Function)

- The logistic function, also known as the sigmoid function, is a key component of logistic regression.
- The sigmoid function maps any real-valued number to a value between 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.
- It has an S-shaped curve, which allows it to model the probability of an instance belonging to a specific class.

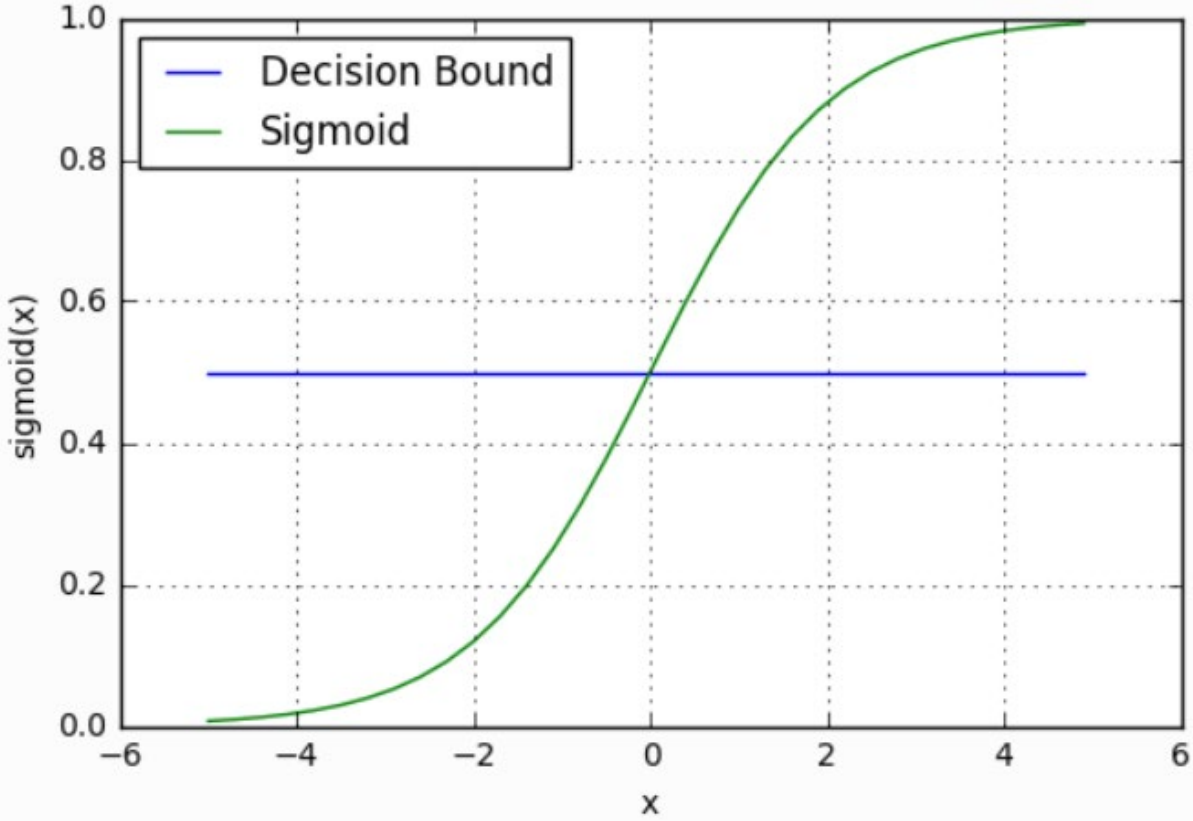
Logistic Function (Sigmoid Function)

- The sigmoid function is defined as follows: f(x) = 1 / (1 + e^(-x)).
- Here, x is the input value, and f(x) represents the output value, which lies between 0 and 1.
- As x approaches positive infinity, f(x) approaches 1, indicating a higher probability of belonging to the positive class.
- Similarly, as x approaches negative infinity, f(x) approaches 0, indicating a lower probability of belonging to the positive class.
- The midpoint of the sigmoid function occurs at x = 0, where f(0) = 0.5, indicating an equal probability of belonging to either class.

Decision Boundary in Logistic Regression

- In a two-dimensional feature space, the decision boundary is represented by a line that separates the positive and negative instances.
- The decision boundary is derived from the sigmoid function output, where the predicted probability is compared to the threshold value.
- Instances with predicted probabilities above the threshold are classified as the positive class, and those below the threshold are classified as the negative class.
- The decision boundary can be linear or non-linear, depending on the relationship between the input variables and the output probability.

Decision Boundary in Logistic Regression



Decision Boundary in Logistic Regression

- Logistic Regression uses a decision boundary to classify instances into different classes based on their predicted probabilities.
- The decision boundary is a threshold value that separates the instances.
- If the predicted probability is above the threshold, the instance is classified as the positive class; otherwise, it is classified as the negative class.
- The decision boundary can be adjusted to control the trade-off between false positives and false negatives

Advantages of Logistic Regression

- Logistic Regression is computationally efficient and can handle large datasets.
- It provides interpretable results by estimating the impact of each input variable on the output probability.
- Logistic Regression can handle both categorical and continuous input variables.
- It is robust to noise and outliers in the data.

Limitations of Logistic Regression

- Logistic Regression assumes a linear relationship between the input variables and the log-odds of the output probability.
- It may not perform well if the classes are not linearly separable.
- It is sensitive to irrelevant or correlated features.
- Logistic Regression is not suitable for multi-class classification without modifications..

Real-world Use Cases of Logistic Regression

- Predicting customer churn in a telecommunications company based on customer demographics and usage patterns.
- Medical diagnosis, such as predicting the likelihood of a patient having a certain disease based on symptoms and test results.
- Credit scoring, determining the probability of default for loan applicants based on their financial information.
- Spam email classification based on the content and metadata of emails.
- Sentiment analysis, classifying the sentiment of customer reviews as positive or negative.

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".

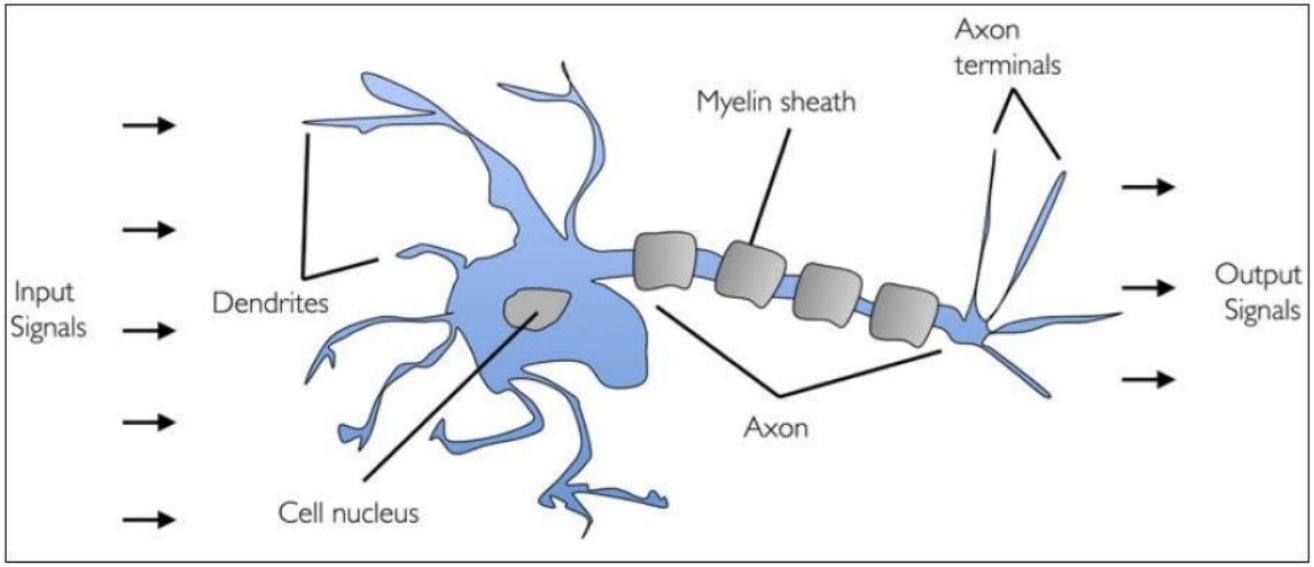
Perceptrons

- A perceptron is a fundamental algorithm in machine learning, serving as the most basic type of artificial neural network. It is a linear binary classifier designed to categorize input data into one of two classes.

- **Biological Neuron**
- A human brain has billions of neurons.
- Neurons are interconnected nerve cells in the human brain that are involved in processing and transmitting chemical and electrical signals.
- Dendrites are branches that receive information from other neurons.
- Cell nucleus or Soma processes the information received from dendrites. Axon is a cable that is used by neurons to send information.
- Synapse is the connection between an axon and other neuron dendrites.

Perceptrons

Biological Neuron



Perceptrons

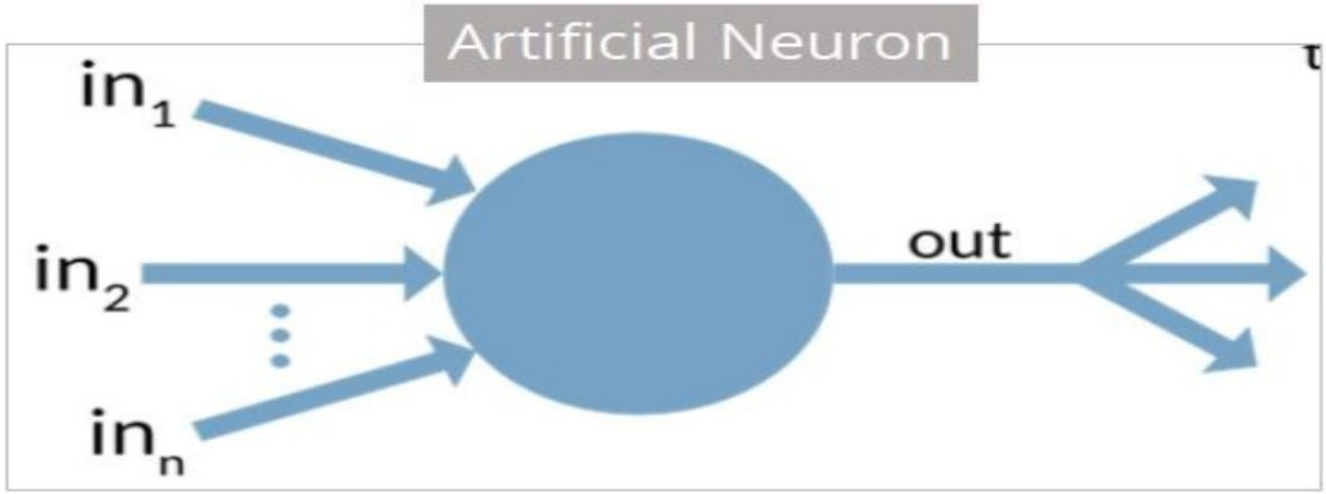
Rise of Artificial Neurons (Based on Biological Neuron)

- Researchers Warren McCullock and Walter Pitts published their first concept of simplified brain cell in 1943. This was called McCullock-Pitts (MCP) neuron.
- They described such a nerve cell as a simple logic gate with binary outputs.
- Multiple signals arrive at the dendrites and are then integrated into the cell body, and, if the accumulated signal exceeds a certain threshold, an output signal is generated that will be passed on by the axon.

Perceptrons

What is Artificial Neuron

An artificial neuron is a mathematical function based on a model of biological neurons, where each neuron takes inputs, weighs them separately, sums them up and passes this sum through a nonlinear function to produce output.



Perceptrons

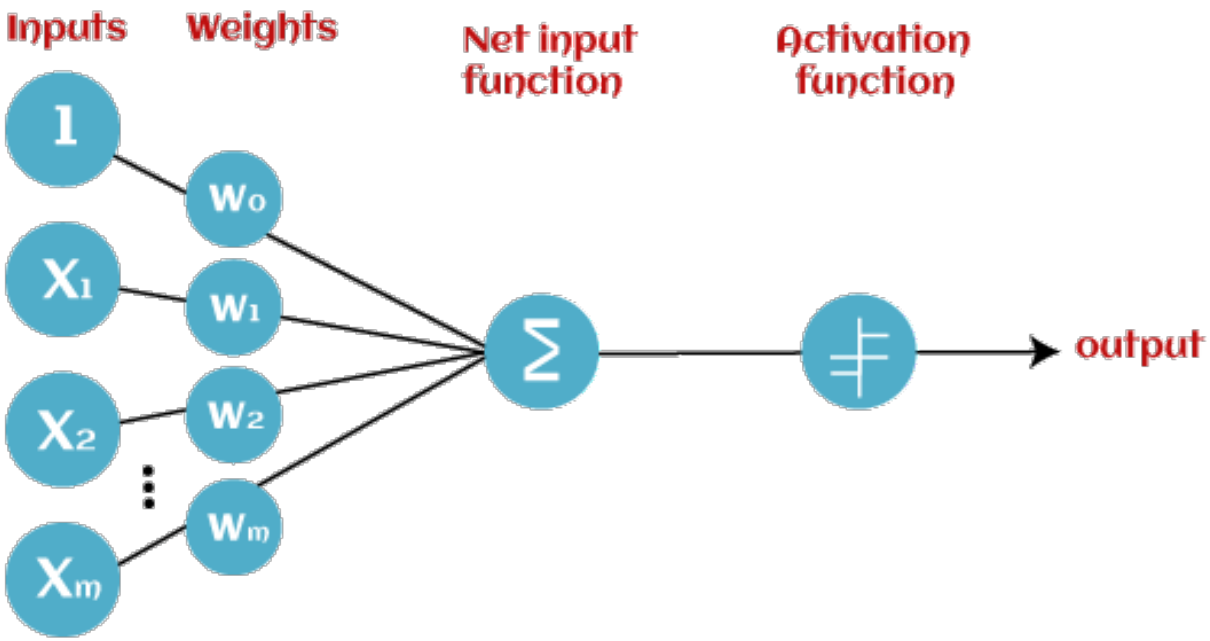
What is Artificial Neuron

- The artificial neuron has the following characteristics:
- A neuron is a mathematical function modeled on the working of biological neurons
- It is an elementary unit in an artificial neural network
- One or more inputs are separately weighted
- Inputs are summed and passed through a nonlinear function to produce output
- Every neuron holds an internal state called activation signal
- Each connection link carries information about the input signal
- Every neuron is connected to another neuron via connection link

Perceptrons

- Perceptron is Machine Learning algorithm for supervised learning of various binary classification tasks.
- Perceptrons are basic building blocks of artificial neural networks and serve as binary classifiers.
- Perceptron model is also treated as one of the best and simplest types of Artificial Neural networks. 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.

Basic Components of Perceptron



Basic Components of Perceptron

- Input Nodes or Input Layer: The input layer consists of one or more input neurons, which receive input signals from the external world or from other layers of the neural network.
- Input units receive input values or features from the dataset.
- Wight :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.
- Bias: A bias term is added to the input layer to provide the perceptron with additional flexibility in modeling complex patterns in the input data. Bias can be considered as the line of intercept in a linear equation.

Basic Components of Perceptron

- Summation Function(net input function): The summation function binds the weights and inputs together. It is a function to find their sum.
- Mathematically,

$$\text{net input} = \sum_{(\text{weight} * \text{input})+\text{bias}}$$

- In general net input =∑2_(*i*=0)^*n*⁡〖*wi*∗*xi*+*b*〗

- The activation function introduces non-linearity and determines the output based on the weighted sum.

Basic Components of Perceptron

- The activation function introduces non-linearity and determines the output based on the weighted sum.
- The activation function introduces non-linearity, allowing perceptrons to learn complex decision boundarie.
- 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.

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0 & \text{otherwise} \end{cases}$$

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.

How does Perceptron work?

Perceptron Example

- Imagine a perceptron (in your brain).
- The perceptron tries to decide if you should go to a concert.
- Is the artist good? Is the weather good?
- What weights should these facts have?

Criteria	Input	Weight
Artists is Good	**x1** = 0 or 1	**w1** = 0.7
Weather is Good	**x2** = 0 or 1	**w2** = 0.6
Friend will Come	**x3** = 0 or 1	**w3** = 0.5
Food is Served	**x4** = 0 or 1	**w4** = 0.3
Alcohol is Served	**x5** = 0 or 1	**w5** = 0.4

How does Perceptron work?

Perceptron Example

- The Perceptron Algorithm: Frank Rosenblatt suggested this algorithm:
- Set a threshold value
- Multiply all inputs with its weights
- Sum all the results
- Activate the output

How does Perceptron work?

Perceptron Example

1. Set a threshold value:
- Threshold = 1.5

2. Multiply all inputs with its weights:
- x1 * w1 = 1 * 0.7 = 0.7
- x2 * w2 = 0 * 0.6 = 0
- x3 * w3 = 1 * 0.5 = 0.5
- x4 * w4 = 0 * 0.3 = 0
- x5 * w5 = 1 * 0.4 = 0.4

3. Sum all the results:
0.7 + 0 + 0.5 + 0 + 0.4 = 1.6 (The Weighted Sum)

4. Activate the Output:
Return true if the sum > 1.5 ("Yes I will go to the Concert")

How does Perceptron work?

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.

- Perceptron is a machine learning algorithm for supervised learning of binary classifiers.
- In Perceptron, the weight coefficient is automatically learned.
- Initially, weights are multiplied with input features, and the decision is made whether the neuron is fired or not.
- The activation function applies a step rule to check whether the weight function is greater than zero.
- The linear decision boundary is drawn, enabling the distinction between the two linearly separable classes +1 and -1.
- 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.

Applications of Perceptron

- Image and pattern recognition: Perceptrons can be used for tasks such as face recognition, object detection, and character recognition.
- Natural language processing: Perceptrons can aid in tasks like sentiment analysis, spam detection, and text classification.
- Financial analysis: Perceptrons can assist in fraud detection, credit scoring, and stock market prediction.
- Medical diagnosis: Perceptrons can be utilized for disease prediction, patient risk assessment, and medical image analysis.
- Robotics and control systems: Perceptrons can be employed for robot navigation, autonomous vehicles, and process control.

Types of Perceptron Models

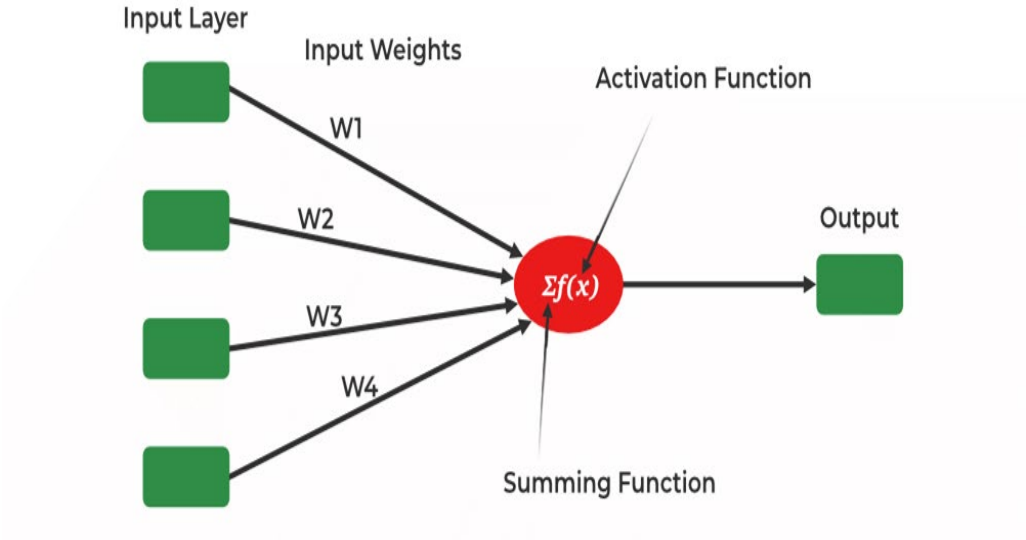
- Based on the layers, Perceptron models are divided into two types. These are as follows:
- Single-layer Perceptron Model
- Multi-layer Perceptron model

Single-layer Perceptron Model –

- It is one of the oldest and first introduced neural networks.
- It was proposed by Frank Rosenblatt in 1958.
- Perceptron is also known as an artificial neural network.
- Perceptron is mainly used to compute the logical gate like AND, OR, and NOR which has binary input and binary output.

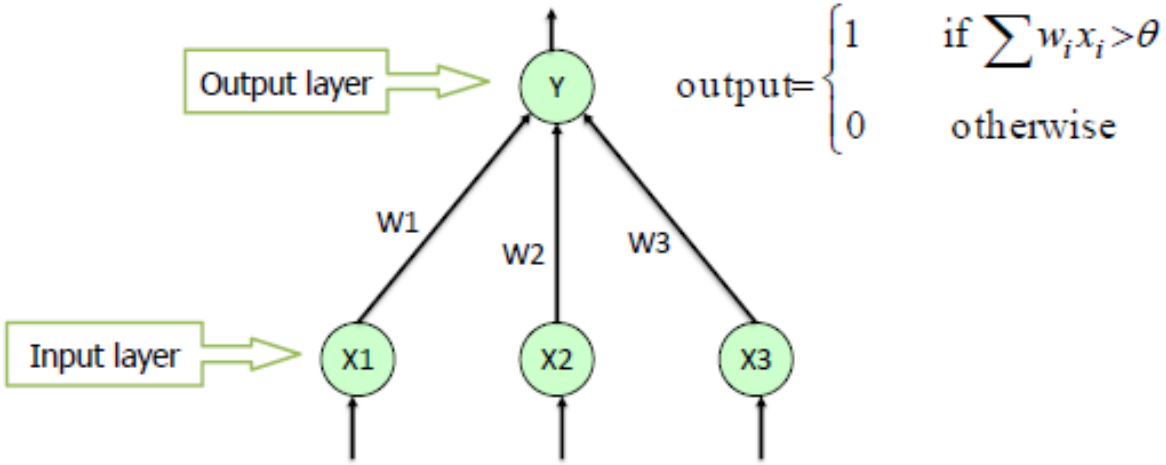
Types of Perceptron Models

- The main functionality of the perceptron is:-
- Takes input from the input layer
- Weight them up and sum it up.
- Pass the sum to the nonlinear function to
- produce the output.



Types of Perceptron Models

Single Layer Perceptron



Types of Perceptron Models

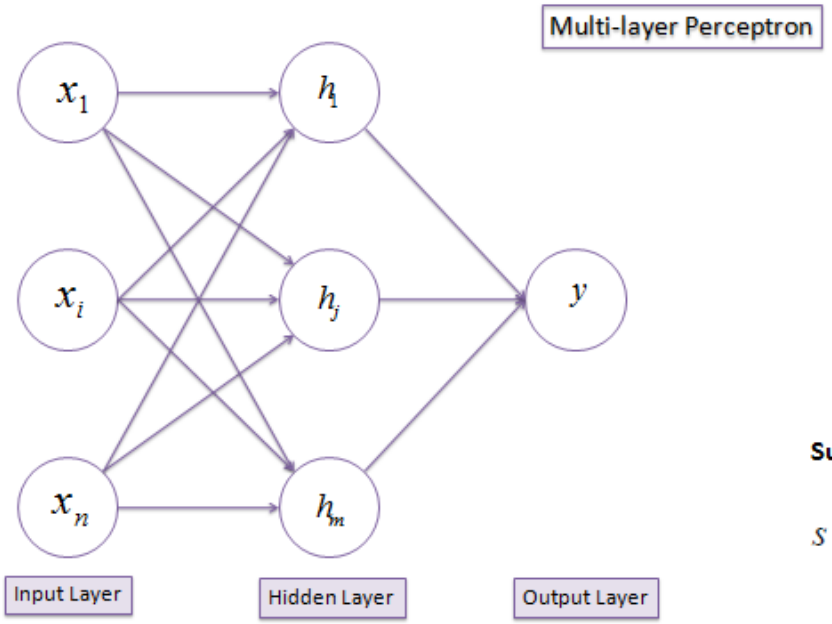
Multi-layer Perceptron model –

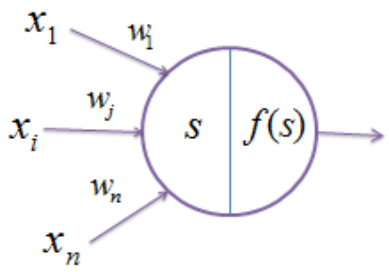
- Like a single-layer perceptron model, a multi-layer perceptron model also has the same model structure but has a greater number of hidden layers.
- The multi-layer perceptron model is also known as the Backpropagation algorithm, which executes in two stages as follows:
- Forward Stage: Activation functions start from the input layer in the forward stage and terminate on the output layer.
- Backward Stage: In the backward stage, weight and bias values are modified as per the model's requirement. In this stage, the error between actual output and demanded originated backward on the output layer and ended on the input layer.

Types of Perceptron Models

Forward propagation –

- Propagate inputs by adding all the weighted inputs and then computing outputs using sigmoid threshold.





Summation

$$s = \sum w \cdot x$$

Transformation

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

Types of Perceptron Models

Backward propagation –

- Propagates the errors backward by apportioning them to each unit according to the amount of this error the unit is responsible for.

1. Error in any ***output*** neuron

$$d_o = y \times (1-y) \times (t-y)$$

2. Error in any ***hidden*** neuron

$$d_i = y_i \times (1-y_i) \times (w_i \times d_o)$$

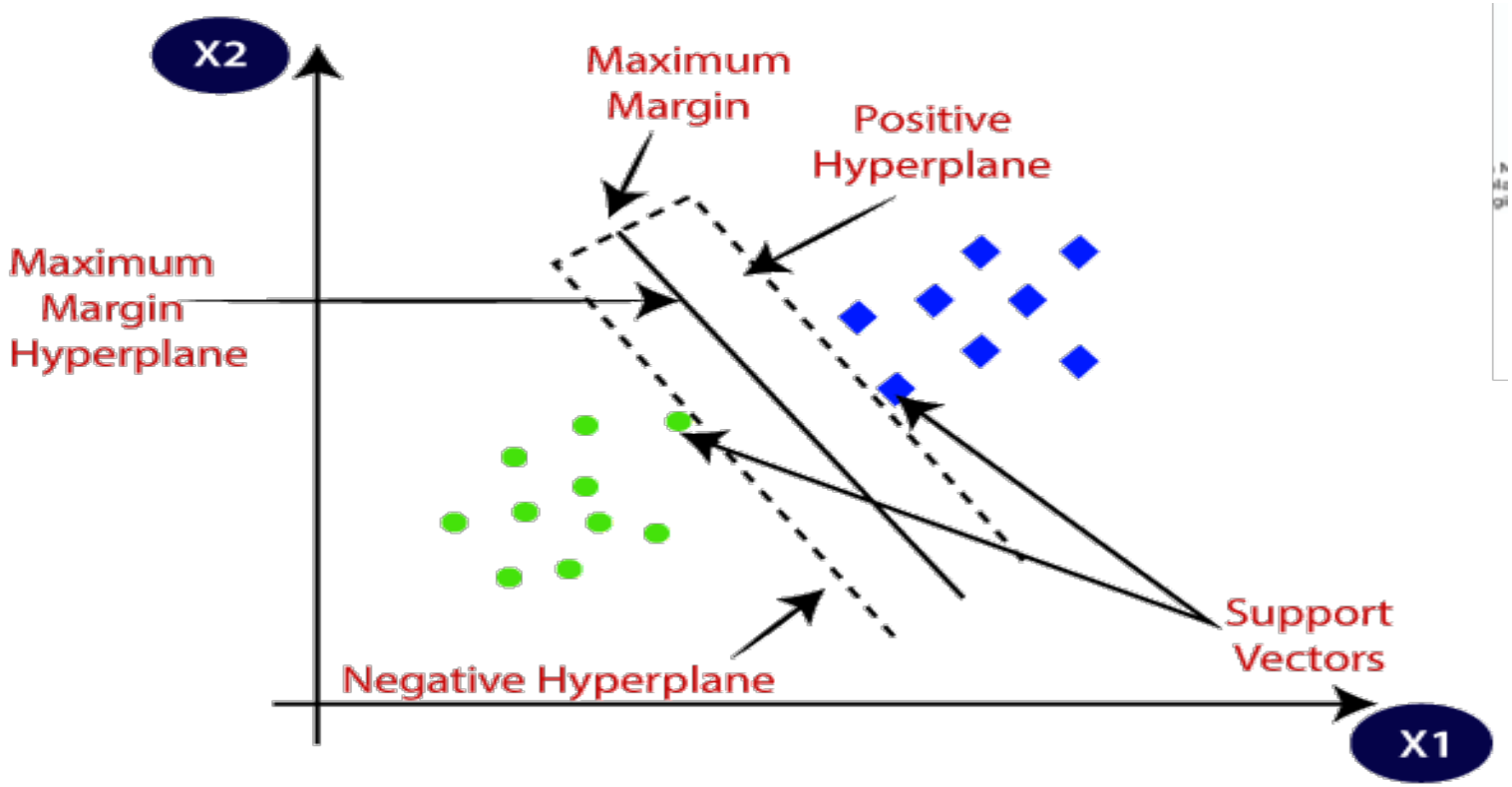
3. Change the ***weights***

$$\Delta w = \eta \times d \times x$$

Support Vector Machine Algorithm

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.
- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
- SVM chooses the extreme points/vectors that help in creating the hyperplane.
- These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.
- Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane.

Support Vector Machine Algorithm

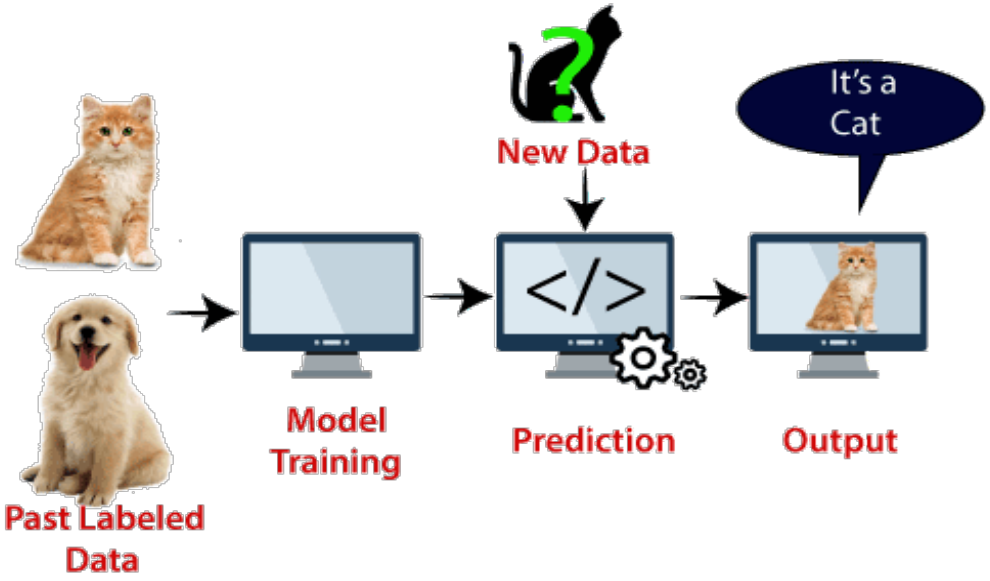


Support Vector Machine Algorithm

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

Support Vector Machine Algorithm

- On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:



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

Types of SVM

SVM can be of two types:

- Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

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

Hyperplane and Support Vectors in the SVM algorithm

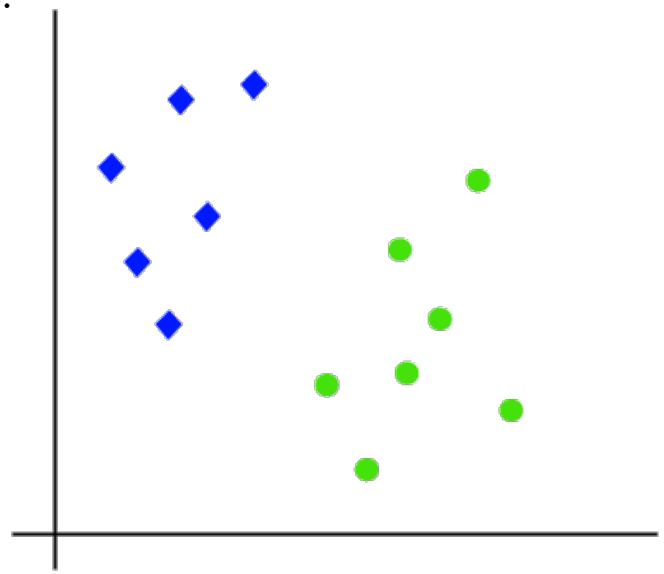
- **Hyperplane**: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.
- The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.
- We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Hyperplane and Support Vectors in the SVM algorithm

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

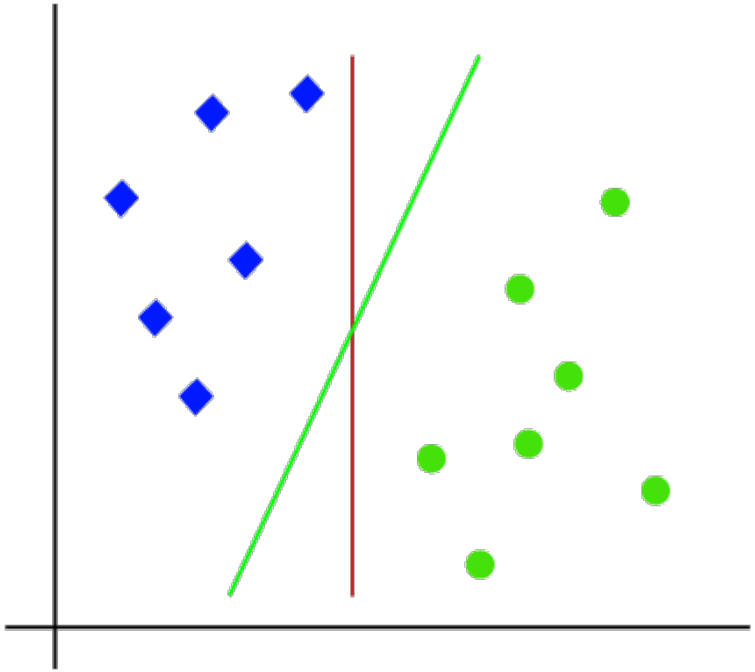
How does SVM works?

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



How does SVM works?

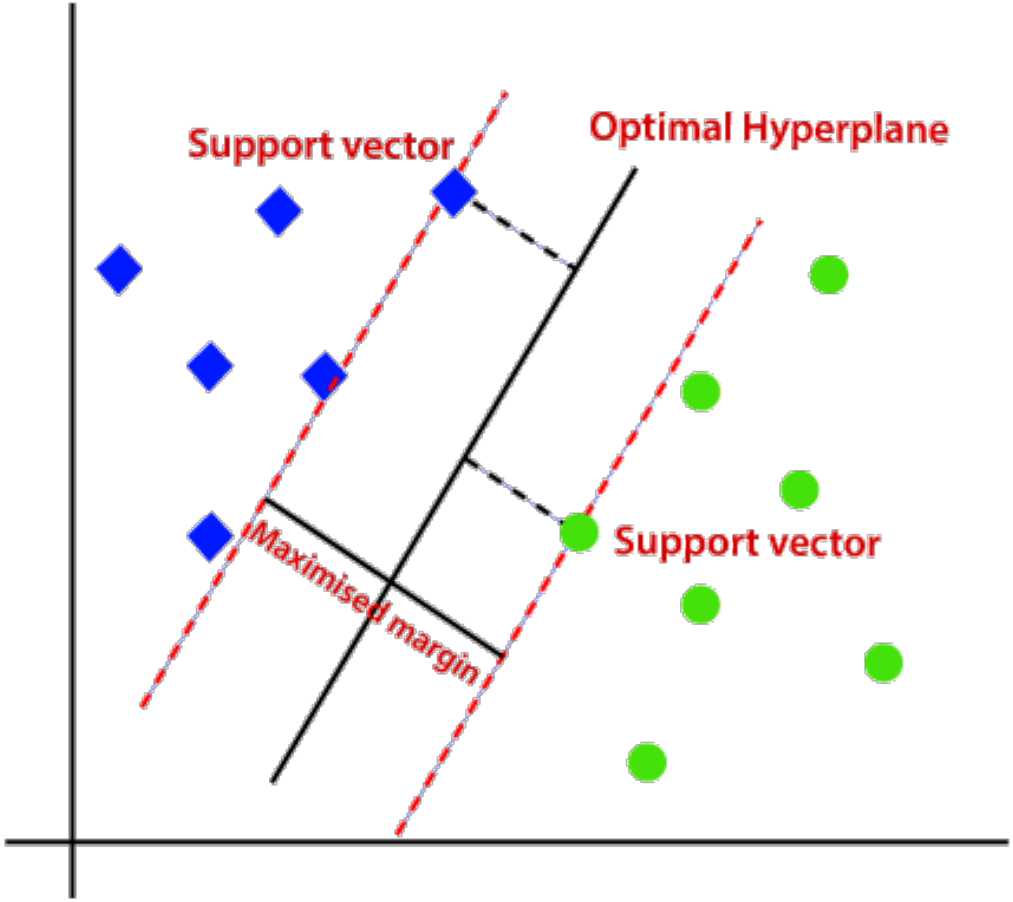
- **Linear SVM:**
- So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:



How does SVM works?

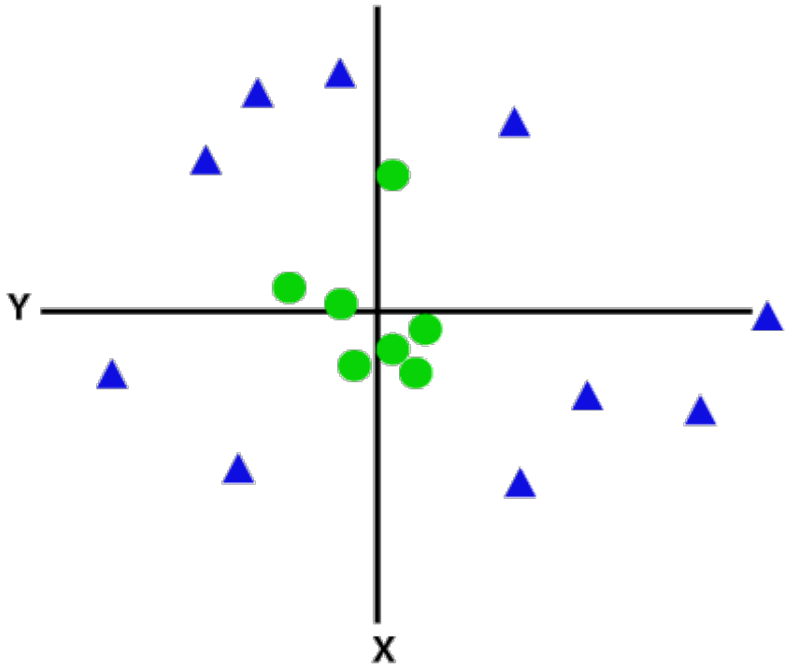
- **Linear SVM:**
- Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a hyperplane. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and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

How does SVM works?



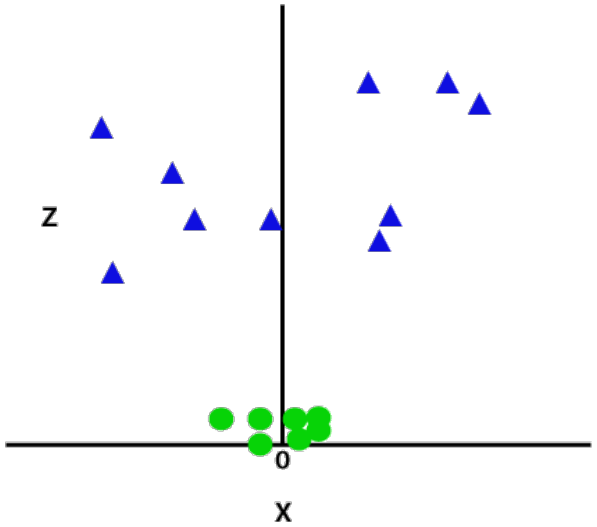
How does SVM works?

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



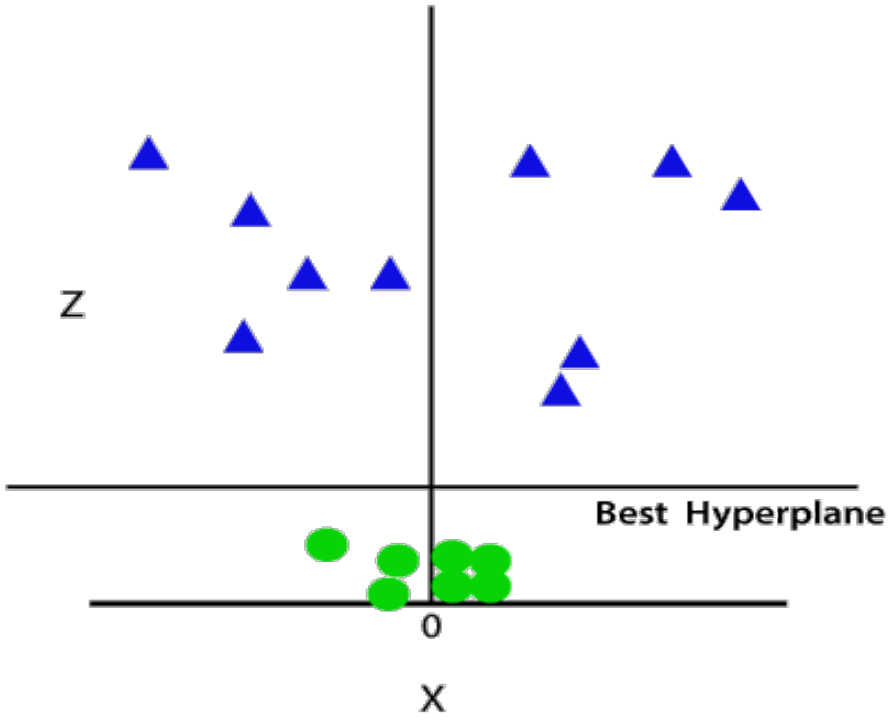
How does SVM works?

- **Non-Linear SVM:**
- 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=x2 +y2
- By adding the third dimension, the sample space will become as shown in image:



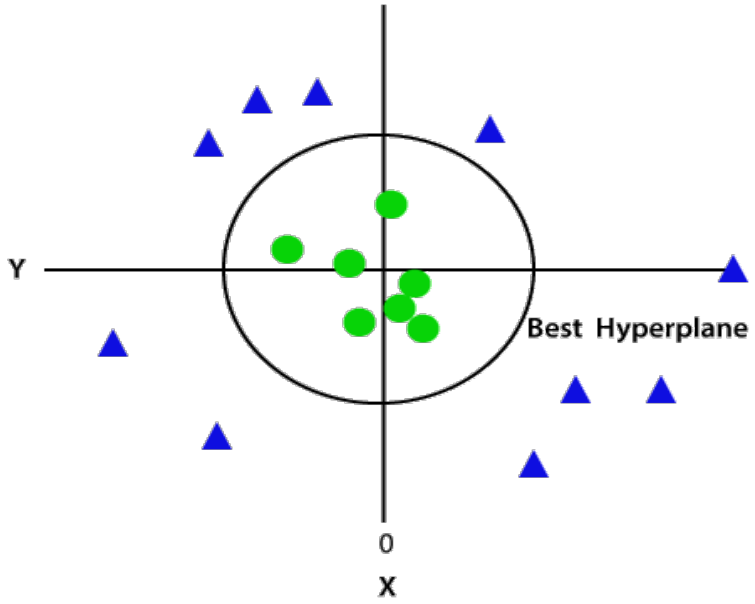
How does SVM works?

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



How does SVM works?

- **Non-Linear SVM:**
- 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.



Parul® University | NAAC GRADE A++

https://paruluniversity.ac.in/