Bayesian Belief Network

Bayesian Belief Network

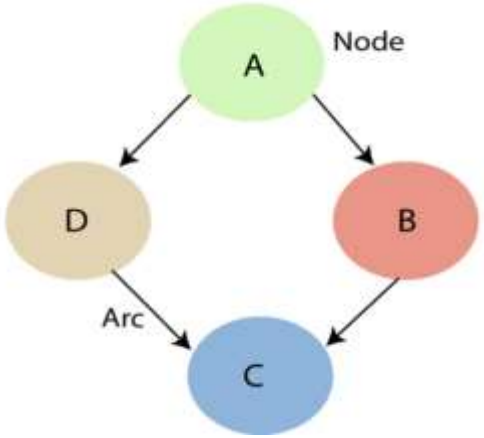
- **Bayesian Belief Network in artificial intelligence**
- Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem which has uncertainty. We can define a Bayesian network as:
- "A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph."
- It is also called a **Bayes network, belief network, decision network**, or **Bayesian model**.
- Bayesian networks are probabilistic, because these networks are built from a **probability distribution**, and also use probability theory for prediction and anomaly detection.

Bayesian Belief Network

- Real world applications are probabilistic in nature, and to represent the relationship between multiple events, we need a Bayesian network. It can also be used in various tasks including **prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction**, and **decision making under uncertainty**.
- Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:
- **Directed Acyclic Graph**
- **Table of conditional probabilities.**

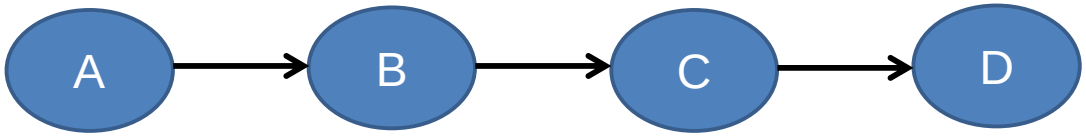
Bayesian Belief Network

- **A Bayesian network graph is made up of nodes and Arcs (directed links), where:**
- Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.
- **Arc or directed arrows** represent the causal relationship or conditional probabilities between random variables. These directed links or arrows connect the pair of nodes in the graph.
  These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other
  - **In the below diagram, A, B, C, and D are random variables represented by the nodes of the network graph.**
  - **If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B.**
  - **Node C is independent of node A.**



Bayesian Belief Network

- **Note: The Bayesian network graph does not contain any cyclic graph. Hence, it is known as a directed acyclic graph or DAG.**



- A directed graph G with four vertices A,B,C, and D. If **p(xA,xB,xC,xD)** factorizes with respect to G, then we must have p(xA,xB,xC,xD) = p(xA)p(xB|xA)p(xC|xB)p(xD|xC).

Bayesian Belief Network

- Bayesian network is based on Joint probability distribution and conditional probability. So let's first understand the joint probability distribution:
- **Joint probability distribution:**
- If we have variables x1, x2, x3,....., xn, then the probabilities of a different combination of x1, x2, x3.. xn, are known as Joint probability distribution.
- **$P[x_1, x_2, x_3,....., x_n]$**, it can be written as the following way in terms of the joint probability distribution.
- **$= P[x_1| x_2, x_3,....., x_n]P[x_2, x_3,....., x_n]$**
- **$= P[x_1| x_2, x_3,....., x_n]P[x_2|x_3,....., x_n]....P[x_{n-1}|x_n]P[x_n].$**
- In general for each variable Xi, we can write the equation as:
- $P(X_i|X_{i-1},....., X_1) = P(X_i |Parents(X_i))$

Explanation Bayesian Belief Network

- Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:
- **Directed Acyclic Graph**
- **Table of conditional probabilities.**

Explanation Bayesian Belief Network

Directed Acyclic Graph: is the pictorial representation of the events that occurred.

Node: Hypothesis

Edge: conditional probability

Note: # node has at least two probabilities.

Probabilities are calculated on the Basis of parent.

Conditional Probability Table

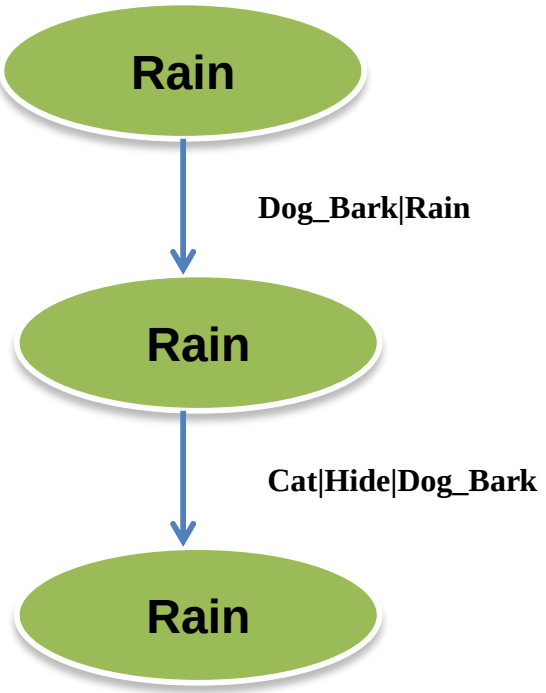
	R	~R
B	9/48	18/48
~B	3/48	18/48

B=T & R=T=0.19

B=T & R=F=0.375

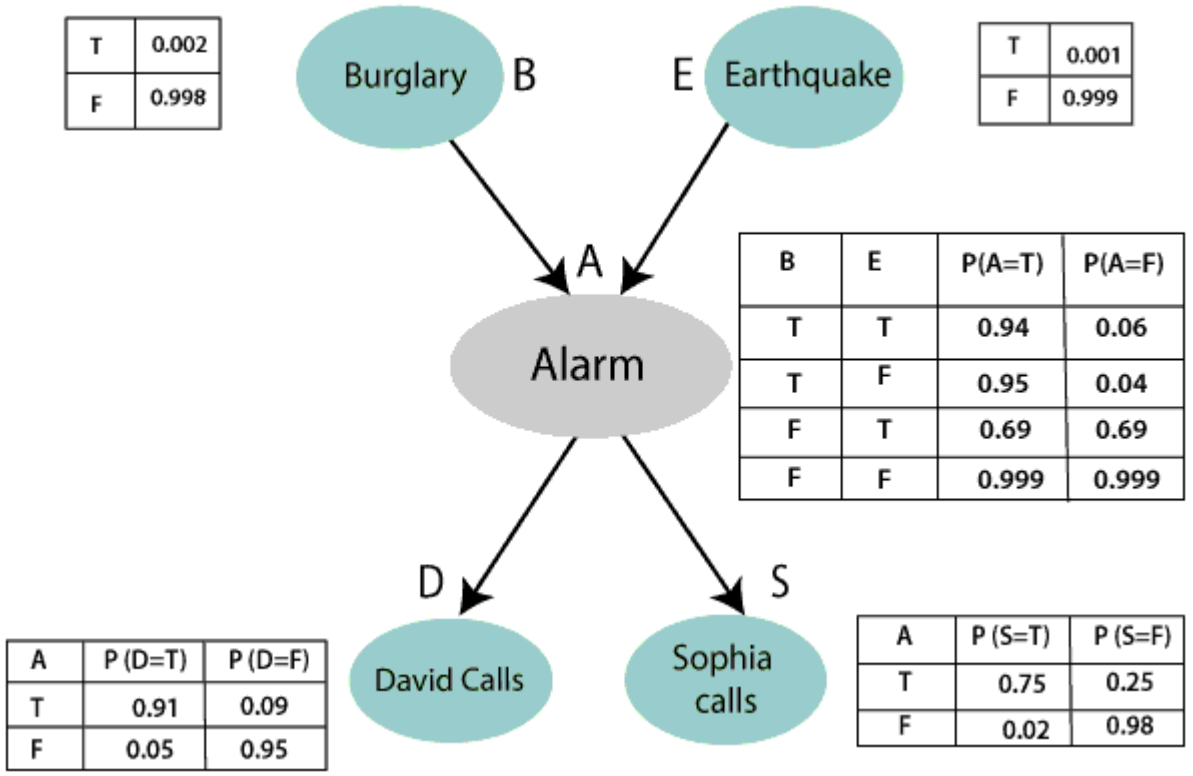
B=F & R=T=0.66

B=F & R=F=0.375



Bayesian Belief Network

- **Solved Example:** Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and David and Sophia both called the Harry.



T	0.002
F	0.998

T	0.001
F	0.999

B	E	P(A=T)	P(A=F)
T	T	0.94	0.06
T	F	0.95	0.04
F	T	0.69	0.69
F	F	0.999	0.999

A	P (D=T)	P (D=F)
T	0.91	0.09
F	0.05	0.95

A	P (S=T)	P (S=F)
T	0.75	0.25
F	0.02	0.98

Bayesian Belief Network

- **P(S, D, A, ¬B, ¬E) = P (S|A) *P (D|A)*P (A|¬B ^ ¬E) *P (¬B) *P (¬E).**
- = 0.75* 0.91* 0.001* 0.998*0.999
- **= 0.00068045.**

Bayesian Modeling

Bayesian Modeling-Statistics

- Classical statistics provides methods to analyze data, from simple descriptive measures to complex and sophisticated models. The available data are processed and then conclusions about a hypothetical population of which the data available are supposed to be a representative sample are drawn.
- Suppose, for example, we need to guess the outcome of an experiment that consists of tossing a coin. How many biased coins have we ever seen? Probably not many, and hence we are ready to believe that the coin is fair and that the outcome of the experiment can be either head or tail with the same probability.
- On the other hand, imagine that someone would tell us that the coin is forged so that it is more likely to land head.
- How can we take into account this information in the analysis of our data?

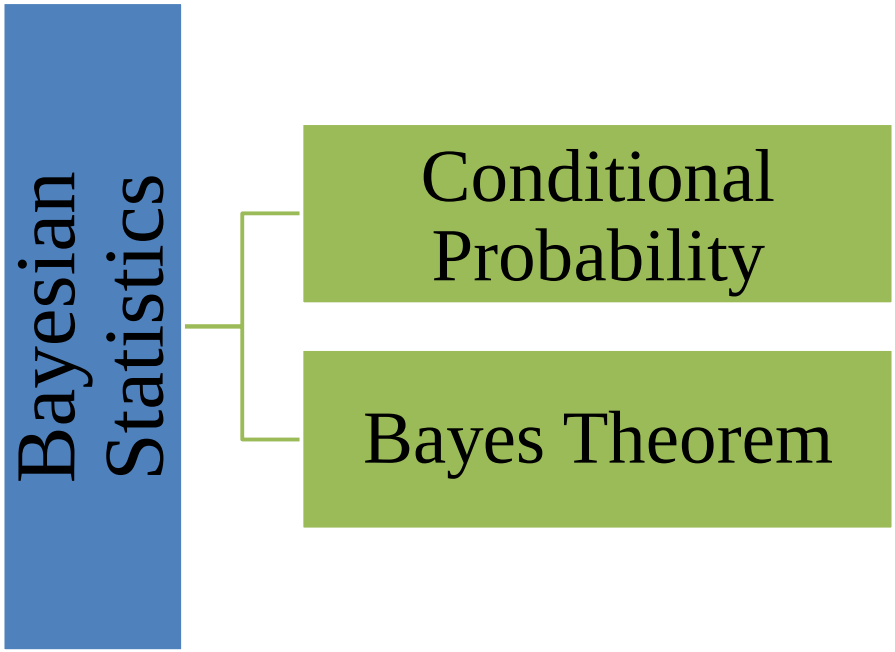
Bayesian Modeling-Statistics

- Bayesian methods provide a principled way to incorporate this external information into the data analysis process. To do so, however, Bayesian methods have to change entirely the vision of the data analysis process with respect to the classical approach. In a Bayesian approach, the data analysis process starts already with a given probability distribution. As this distribution is given *before* any data is considered, it is called *prior* distribution.

Bayesian Modeling-Statistics

- Bayesian statistics is a mathematical procedure that applies probabilities to statistical problems. It provides people the tools to update their beliefs in the evidence of new data.
- Suppose, out of all the 4 championship races (F1) between Niki Lauda and James hunt, Niki won 3 times while James managed only 1.
- So, if you were to bet on the winner of next race, who would he be ?
- I bet you would say Niki Lauda.
- Here's the twist. What if you are told that it rained once when James won and once when Niki won and it is definite that it will rain on the next date. So, who would you bet your money on now ?
- By intuition, it is easy to see that chances of winning for James have increased drastically. But the question is: how much ?

Bayesian Modeling-Statistics



Conditional Probability

- **Conditional probability** is known as the possibility of an event or outcome happening, based on the existence of a previous event or outcome. It is calculated by multiplying the probability of the preceding event by the renewed probability of the succeeding, or conditional, event.
- The probability of occurrence of any event A when another event B in relation to A has already occurred is known as conditional probability. It is depicted by P(A|B).

Conditional Probability

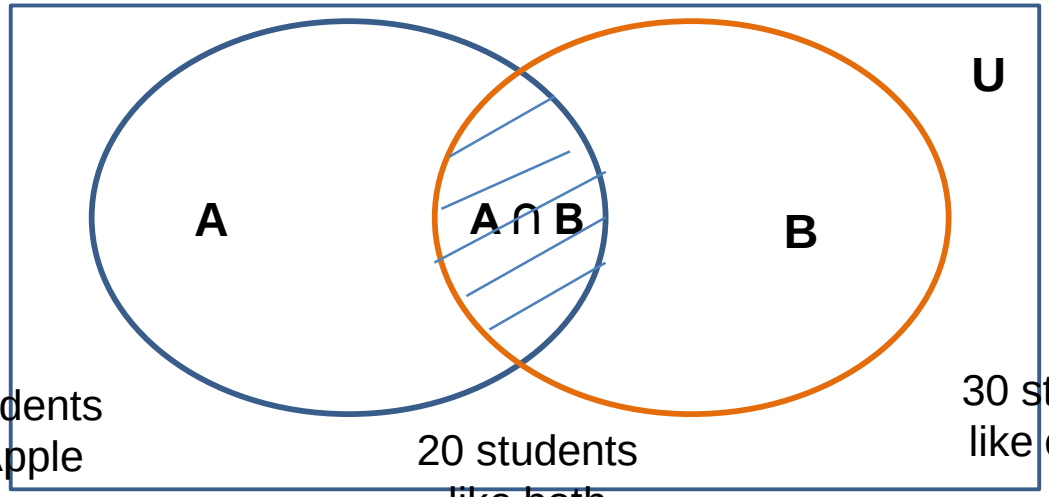
Conditional Probability Formula

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Probability of A given B

Probability of A and B

Probability of B



Conditional Probability

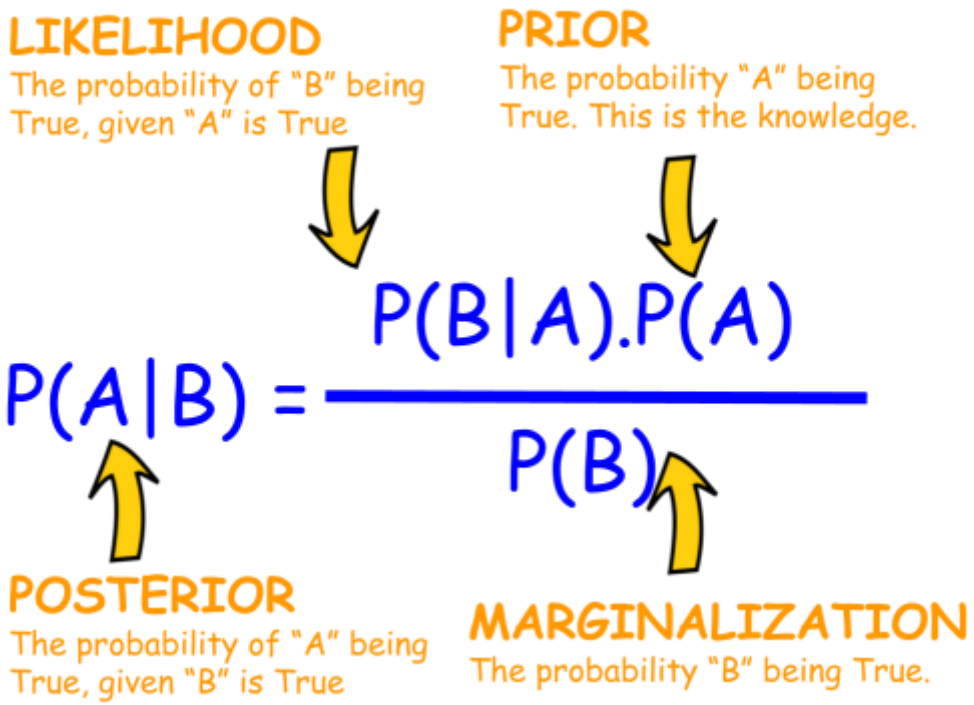
$$P(B) = \frac{30}{100} = 0.3$$

$$P(A \cap B) = \frac{20}{100} = 0.2$$

$$P(A \mid B) = \frac{0.2}{0.3} = 0.67$$

Bayes Theorem

- Bayes''theorem is a mathematical formula used to determine the conditional probability of the events.
- Bayes theorem describes the probability of an event based on prior knowledge of the conditions that might be relevant to the event.
- Invented – Thomas Bayes
- Year- 1763



Posterior P(A|B)=> Probability of event A being True, given event B has already accrued.

Likelihood P(B|A)=> Probability of the evidence given that the hypothesis is True.

Prior P(A)=> Probability of hypothesis before considering the evidence.

Marginal P(B)=> Probability of evidence/Data.

Bayes Theorem

- P(A|B)- Probability of hypothesis A, given that evidence or data B.
- P(B|A)- Probability of data/evidence, given that hypothesis is true.
- P(A)- Probability of A
- P(B)- Probability of B

$$P(A \mid B) = \frac{P \cap B}{P(B)}$$

$$P(B \mid A) = \frac{P(B \cap A)}{P(A)}$$

$$LHS \qquad \text{RHS}$$

$$P(A \mid B).P(B) = P(A \cap B)$$

$$P(B \mid A).P(A) = P(B \cap A)$$

$$P(A \cap B) = P(A \mid B).P(B) = P(B \mid A).P(A)$$

$$P(A \mid B) = \frac{P(B \mid A).P(A)}{P(B)}$$

Bayes Theorem

- For Example:
- Calculate P(King|Face)------Posterior Probability

$$P(King \mid Face) = \frac{(Face \mid King).P(King)}{P(Face)}$$

$$P(King \mid Face) = \frac{(1).(4/52)}{12/52}$$

$$P(King \mid Face) = \frac{(1).(1/13)}{(3/13)} = \frac{1}{3} = 0.33$$

Naïve Bayes

- Naive Bayes algorithm is a classification technique based on Bayes' theorem, which assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. There are various applications of this algorithm including face recognition, NLP problems, medical diagnoses and a lot more.

Naive Bayes example:

Below is training data on which Naive Bayes algorithm is applied:

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Step 1: Make a Frequency table of the data.

Frequency Table		
Weather	**No**	**Yes**
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Step 2: Create a Likelihood table by finding probabilities like Overcast probability = 0.29.

Frequency Table				
Weather	**No**	**Yes**		
Overcast		4	**=4/14**	**0.29**
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
Grand Total	5	9		
	=5/14	**=9/14**		
	0.36	0.64		

Step 3: Use Naive Bayes equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Problem: Players will play if the weather is Rainy. Is this statement correct?

You can solve it using the above discussed method of posterior probability.

P(Yes | Rainy) = P(Rainy | Yes) * P(Yes) / P (Rainy)

Here, you have P (Rainy |Yes) = 2/9 = 0.22, P(Rainy) = 5/14 = 0.36, P(Yes)= 9/14 = 0.64

Now, P (Yes | Rainy) = 0.22 * 0.64 / 0.36 = 0.39, which has a higher probability.

Naive Bayes uses a similar method to predict the probability of different classes based on various attributes. This algorithm is mostly used in NLP problems like sentiment analysis, text classification, etc.

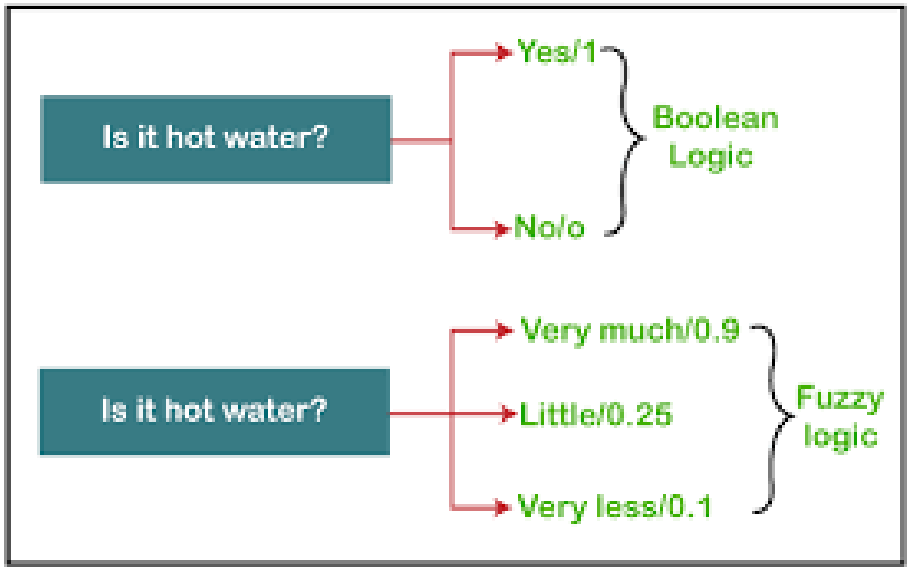
Fuzzy Logic

Fuzzy

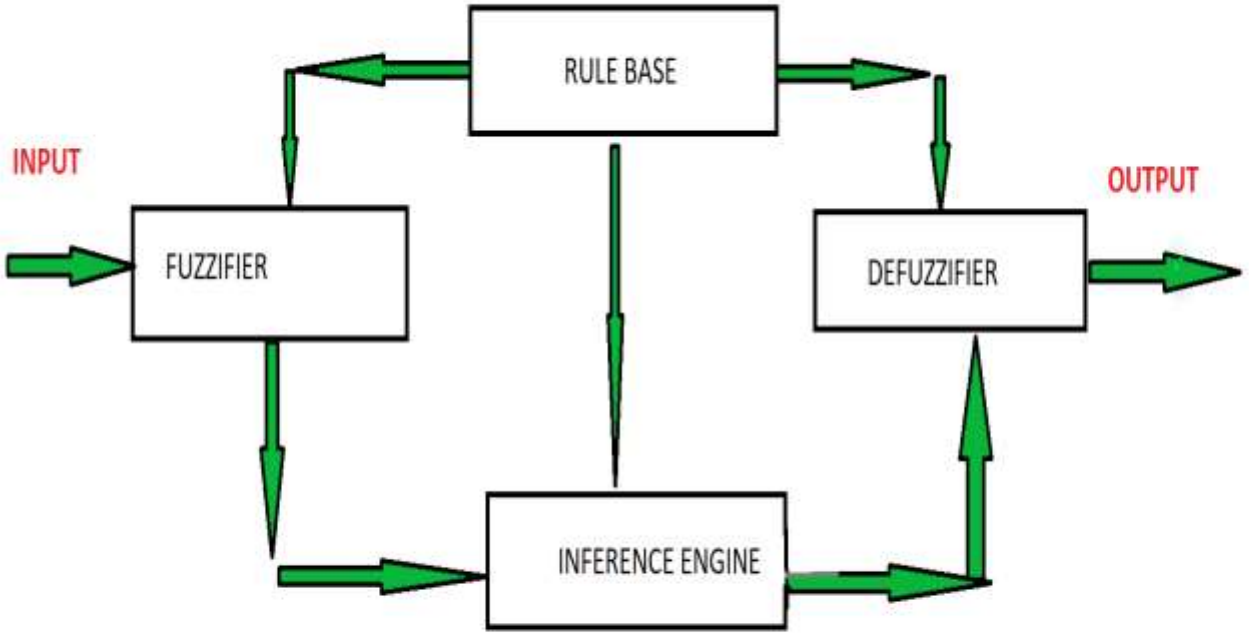
- The word **fuzzy** refers to things which are not clear or are vague. Any event, process, or function that is changing continuously cannot always be defined as either true or false, which means that we need to define such activities in a Fuzzy manner.

Fuzzy-Logic

- Fuzzy Logic resembles the human decision-making methodology. It deals with vague and imprecise information. This is gross oversimplification of the real-world problems and based on degrees of truth rather than usual true/false or 1/0 like Boolean logic.
- Take a look at the following diagram. It shows that in fuzzy systems, the values are indicated by a number in the range from 0 to 1. Here 1.0 represents **absolute truth** and 0.0 represents **absolute falseness**. The number which indicates the value in fuzzy systems is called the **truth value**.



Fuzzy Architecture



FUZZY LOGIC ARCHITECTURE

Fuzzy-Architecture

- **RULE BASE:** It contains the set of rules and the IF-THEN conditions provided by the experts to govern the decision-making system, on the basis of linguistic information. Recent developments in fuzzy theory offer several effective methods for the design and tuning of fuzzy controllers. Most of these developments reduce the number of fuzzy rules.
- **FUZZIFICATION:** It is used to convert inputs i.e. crisp numbers into fuzzy sets. Crisp inputs are basically the exact inputs measured by sensors and passed into the control system for processing, such as temperature, pressure, rpm's, etc.
- **INFERENCE ENGINE:** It determines the matching degree of the current fuzzy input with respect to each rule and decides which rules are to be fired according to the input field. Next, the fired rules are combined to form the control actions.
- **DEFUZZIFICATION:** It is used to convert the fuzzy sets obtained by the inference engine into a crisp value. There are several defuzzification methods available and the best-suited one is used with a specific expert system to reduce the error.

U : All Students

G: Good Students

S: Bad Students

G= {G, $\mu(G)$} $\mu()$- Degree of Goodness

G={(A, 0.9), (B, 0.7), (C, 0.1), (D, 0.3)}

S ={(A, 0.1), (B,0.3), (C, 0.9) (D, 0.7)}

Membership Function

- A membership function for a fuzzy set A on the universe of discourse X is defined as $\mu A : X \rightarrow [0,1]$, where each element of X is mapped to a value between 0 and 1.
- This value, called membership value or degree of membership, quantifies the grade of membership of the element in X to the fuzzy set A.
- Membership functions characterize fuzziness (*i.e.*, all the information in fuzzy set), whether the elements in fuzzy sets are discrete or continuous.
- Membership functions can be defined as a technique to solve practical problems by experience rather than knowledge.
- Membership functions are represented by graphical forms.

Feature of The Membership Function

- **Core :**
- The core of a membership function for some fuzzy set is defined as that region of the universe that is characterized by complete and full membership in the set.
- b. The core comprises those elements *x* of the universe such that

$$\mu_A(x) = 1$$

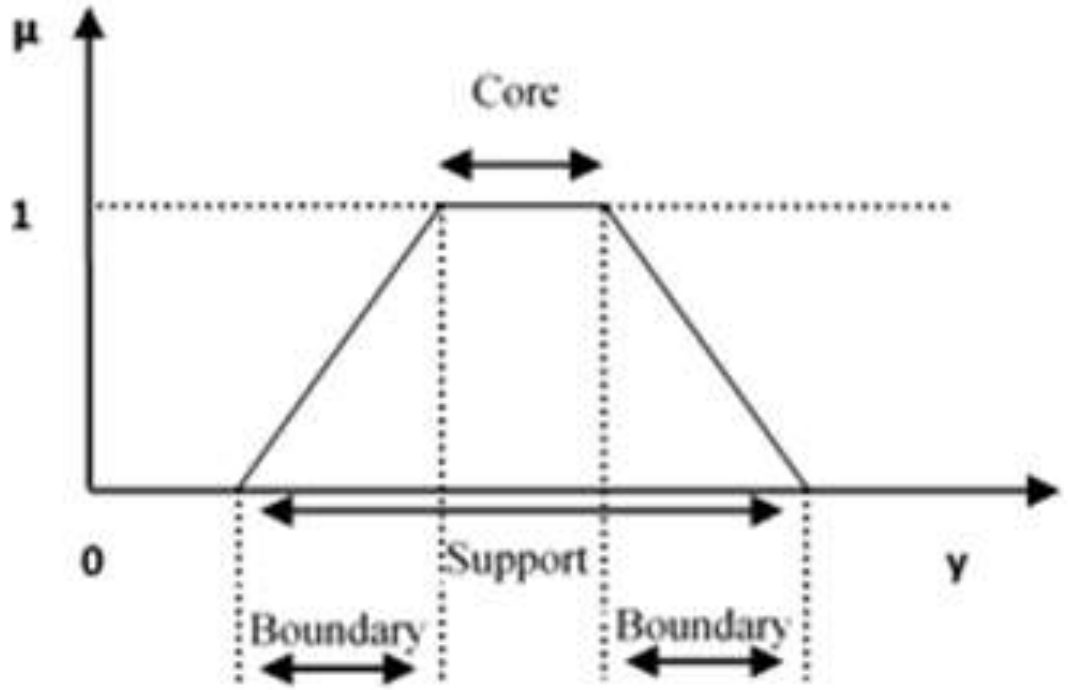
- **Support :**
- a. The support of a membership function for some fuzzy set *A* is
- defined as that region of the universe that is characterized by nonzero membership in the set *A*.
- b. The support comprises those elements *x* of the universe such that

$$\mu_A(x) > 1$$

Feature of The Membership Function

- **3. Boundaries :**
- The boundaries of a membership function for some fuzzy set are defined as that region of the universe containing elements that have a non-zero membership but not complete membership.
- The boundaries comprise those elements *x* of the universe such that

$$0 < \mu_A(x) < 1$$



Features of Membership Function

Benefits of Fuzzy Logic in Real Life

- 1. Fuzzy logic is essential for the development of human-like capabilities for AI.
- 2. It is used in the development of intelligent systems for decision making, identification, optimization, and control.
- 3. Fuzzy logic is extremely useful for many people involved in research and development including engineers, mathematicians, computer software developers and researchers.
- 4. Fuzzy logic has been used in numerous applications such as facial pattern recognition, air conditioners, vacuum cleaners, weather forecasting systems, medical diagnosis and stock trading.

Fuzzy Decision Tree

Decision trees are one of the most popular methods for learning and reasoning from instances.

2. Given a set of n input-output training patterns $D = \{(X_i, y_i)\ i = 1, \ldots, n\}$.

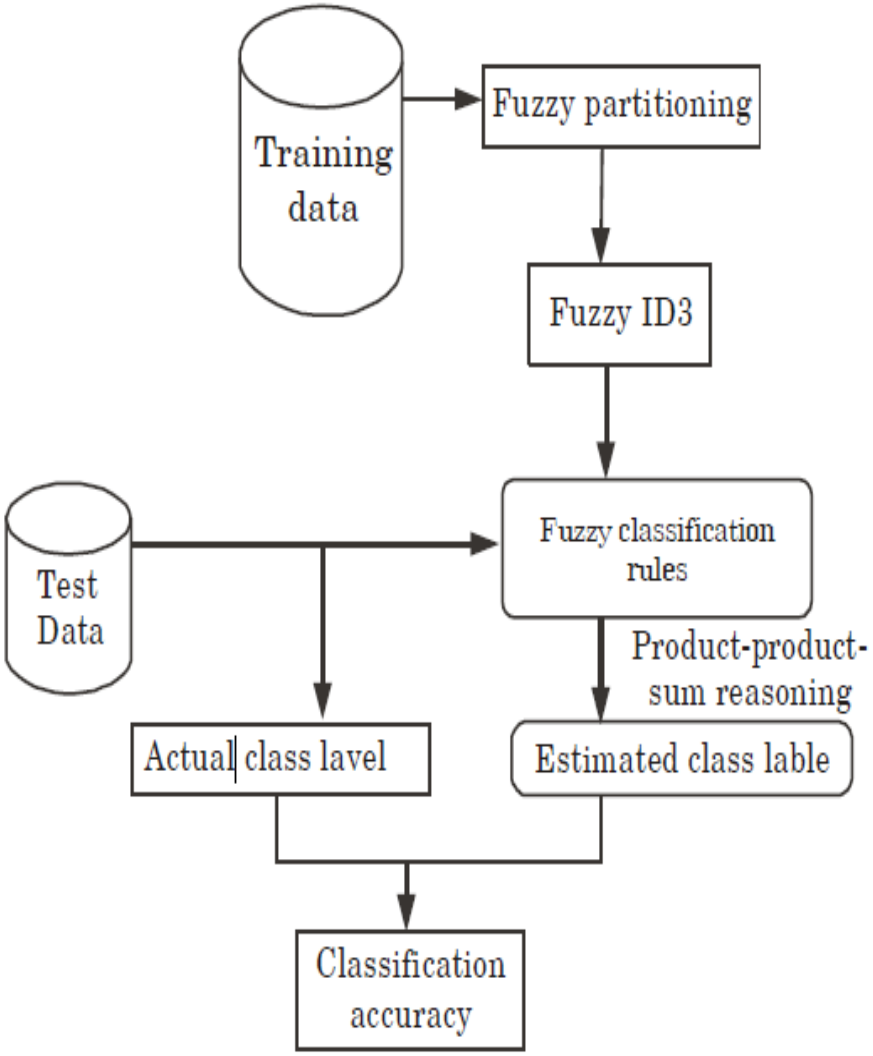
where each training pattern X_i has been described by a set of p conditional (or input) attributes $(x_1,\ldots,x_p)$ and one corresponding discrete class label y_i where $y^i \in \{1,\ldots\ldots,q\}$ and q is the number of classes.

3. The decision attribute y_i represents a posterior knowledge regarding the class of each pattern.

4. An arbitrary class has been indexed by $l (1 \le l \le q)$ and each class l has been modeled as a crisp set.

5. The membership degree of the i_{th} value of the decision attribute y_i concerning the i_{th} class is defined as follows:

$$\mu_i(y^i) = \begin{cases} 1, \text{ if } y^i \text{ belong to } l^{th} class; \\ 0, \text{ otherwise.} \end{cases}$$



1. The generation of FDT for pattern classification consists of three major steps namely fuzzy partitioning (clustering), induction of FDT and fuzzy rule inference for classification.
2. The first crucial step in the induction process of FDT is the fuzzy partitioning of input space using any fuzzy clustering techniques.
3. FDTs are constructed using any standard algorithm like Fuzzy ID3 where we follow a top-down, recursive divide and conquer approach, which makes locally optimal decisions at each node.
4. As the tree is being built, the training set is recursively partitioned into smaller subsets and the generated fuzzy rules are used to predict the class of an unseen pattern.

Stochastic Search Methods

- Stochastic optimization algorithms provide an alternative approach that permits less optimal local decisions to be made within the search procedure that may increase the probability of the procedure locating the global optima of the objective function.
- Stochastic search and optimization pertains to problems where there is randomness noise in the measurements provided to the algorithm and/or there is injected (Monte Carlo) randomness in the algorithm itself.
- Methods for stochastic optimization provide a means of coping with inherent system noise and coping with models or systems that are highly nonlinear, high dimensional, or otherwise inappropriate for classical deterministic methods of optimization.
- The use of randomness in the algorithms often means that the techniques are referred to as “heuristic search” as they use a rough rule-of-thumb procedure that may or may not work to find the optima instead of a precise procedure.
- Many stochastic algorithms are inspired by a biological or natural process and may be referred to as “Metaheuristic” as a higher-order procedure providing the conditions for a specific search of the objective function. They are also referred to as “black box” optimization algorithms.

Types of Stochastic Search Methods

- Stochastic Hill Climbing
- Stochastic Gradient Descent
- Tabu Search

Hill Climbing Algorithm

- Simple hill climbing is the simplest way to implement a hill climbing algorithm. **It only evaluates the neighbor node state at a time and selects the first one which optimizes current cost and set it as a current state**. It only checks it's one successor state, and if it finds better than the current state, then move else be in the same state. This algorithm has the following features:
- Less time consuming
- Less optimal solution and the solution is not guaranteed

➢ **Stochastic Hill-Climbing:** Stochastic hill climbing does not examine for all its neighbor before moving. Rather, this search algorithm selects one neighbor node at random and decides whether to choose it as a current state or examine another state.

Stochastic Gradient Decent

- Gradient Descent is a generic optimization algorithm capable of finding optimal solutions to a wide range of problems.
- The general idea is to tweak parameters iteratively in order to minimize the cost function.
- An important parameter of Gradient Descent (GD) is the size of the steps, determined by the learning rate hyperparameters. If the learning rate is too small, then the algorithm will have to go through many iterations to converge, which will take a long time, and if it is too high we may jump the optimal value.
- The word '*stochastic*' means a system or process linked with a random probability. Hence, in Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration. In Gradient Descent, there is a term called "batch" which denotes the total number of samples from a dataset that is used for calculating the gradient for each iteration. In typical Gradient Descent optimization, like Batch Gradient Descent, the batch is taken to be the whole dataset. Although using the whole dataset is really useful for getting to the minima in a less noisy and less random manner, the problem arises when our dataset gets big.

Tabu Search

- Tabu Search is a commonly used meta-heuristic used for optimizing model parameters. A meta-heuristic is a general strategy that is used to guide and control actual heuristics. Tabu Search is often regarded as integrating **memory structures** into **local search strategies**.
- The basic idea of Tabu Search is to penalize moves that take the solution into previously visited search spaces (also known as **tabu**).

Tabu Search Algorithm

- **Step 1:** We first start with an initial solution ***s*** = ***S₀***. This can be any solution that fits the criteria for an acceptable solution.
- **Step 2:** Generate a set of neighbouring solutions to the current solution ***s*** labeled ***N(s)***.
- **Step 3:** Choose the best solution out of ***N(s)*** and label this new solution ***s'***. If the solution ***s'*** is better than the current best solution, update the current best solution. After, regardless if ***s'*** is better than ***s***, we update ***s*** to be ***s'***.
- **Step 4:** Update the Tabu List ***T(s)*** by removing all moves that are expired past the Tabu Tenure and add the new move ***s'*** to the Tabu List. Additionally, update the set of solutions that fit the Aspiration Criteria ***A(s)***. If frequency memory is used, then also increment the frequency memory counter with the new solution.
- **Step 5:** If the Termination Criteria are met, then the search stops or else it will move onto the next iteration. Termination Criteria is dependent upon the problem at hand but some possible examples are:
- a max number of iterations.
- if the best solution found is better than some threshold.

Artificial Neural Network

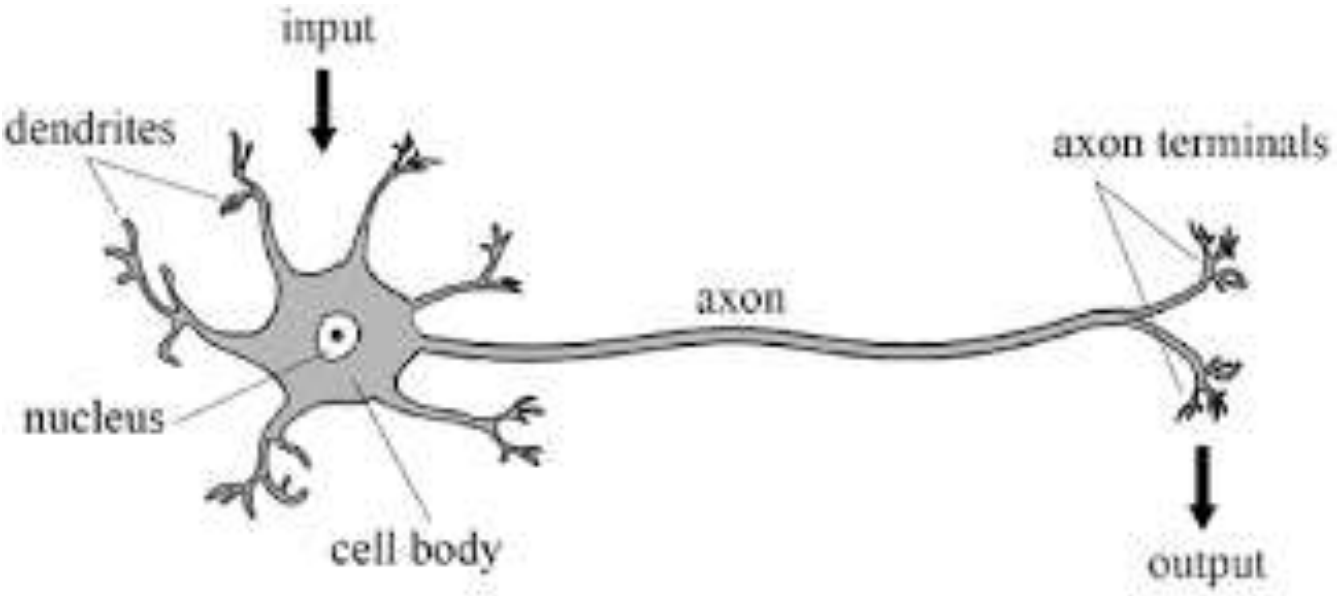
Introduction

- Artificial Neural Networks (ANN) are algorithms based on brain function and are used to model complicated patterns and forecast issues. The Artificial Neural Network (ANN) is a deep learning method that arose from the concept of the human brain Biological Neural Networks. The development of ANN was the result of an attempt to replicate the workings of the human brain. The workings of ANN are extremely similar to those of biological neural networks, although they are not identical. ANN algorithm accepts only numeric and structured data.

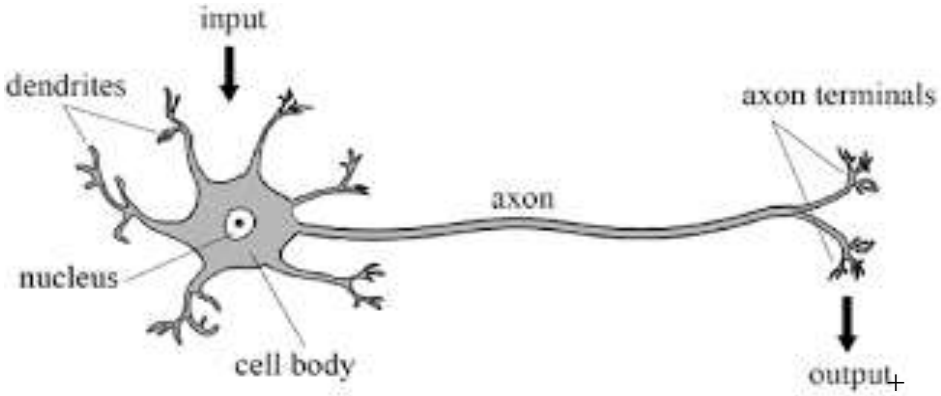
- ANN are biological inspired simulations performed on computer to perform certain tasks like- clustering, classification, pattern recognition.

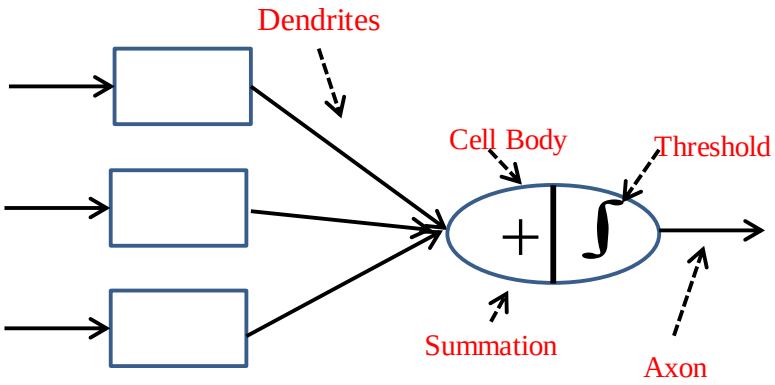
What is neuron network

- The term ‘Neural’ is derived from the human neuron systems, basic functional unit ‘neuron’ or nerve cell that are presents in the brain and other parts of human body.



What is neuron network

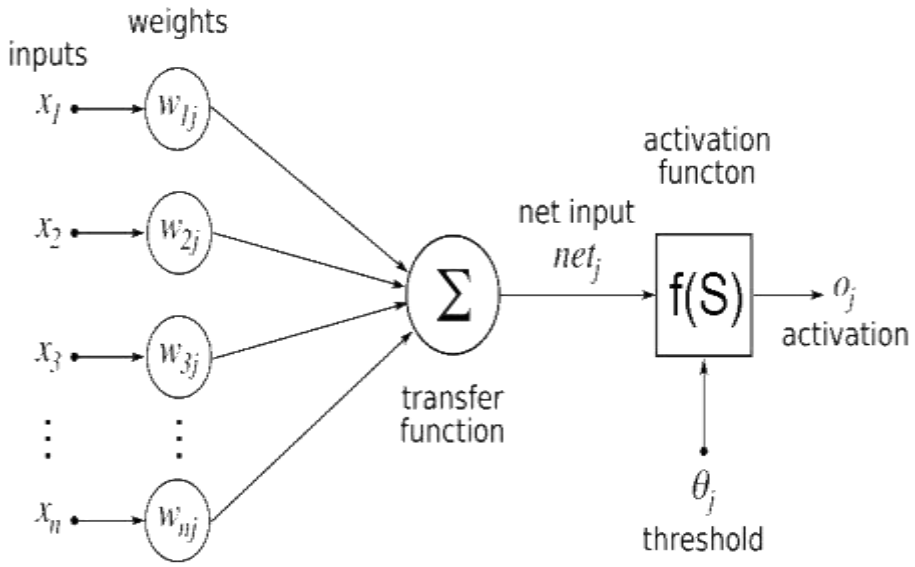




- **Dendrites:** Receives signal from other neurons.
- **Soma/Cell Body:** It seems all the "incoming" signals to generate outputs.
- **Axon Structure:** When the sum reaches to the threshold value, neuron fires and the signal travel to axon to other neuron.
- **Semapses/Axon Terminals:** The point of intersection of one neuron with other neuron. The amount of signal transmitted depends on the strength (weights) of the connections.

How Does ANN Works?

- Artificial Neural Networks work in a way similar to that of their biological inspiration. They can be considered as weighted directed graphs where the neurons could be compared to the nodes and the connection between two neurons as weighted edges. The processing element of a neuron receives many signals (both from other neurons and as input signals from the external world).
- Signals are sometimes modified at the receiving synapse and the weighted inputs are summed at the processing element. If it crosses the threshold, it goes as input to other neurons (or as output to the external world) and the process repeats.

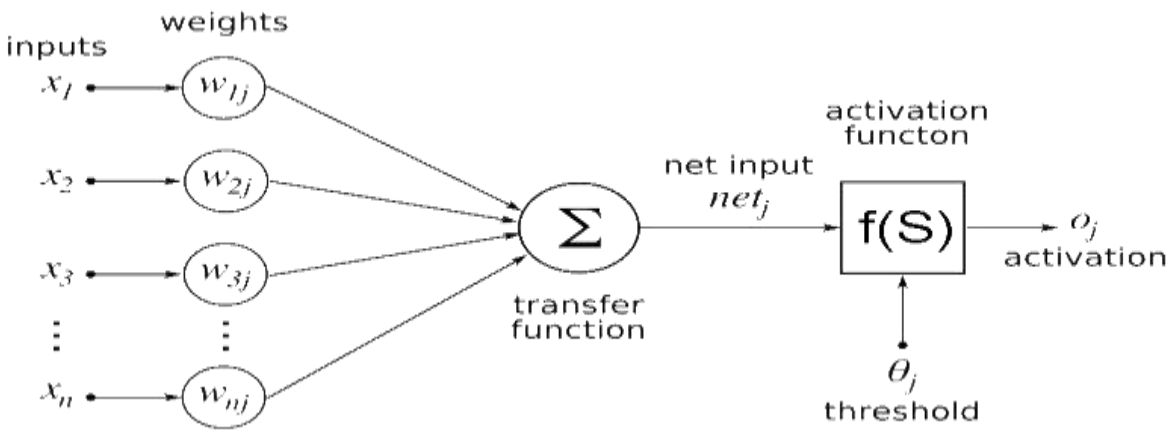


- **weights** represent the interconnection strength between the neurons.
- **activation** function is a transfer function that is used to get the desired output for the problem designed.
- Say, the desired output is zero or one in case of a binary classifier. Sigmoid function could be used as the activation function.

Neuron Calculation

- The equation for the neural network is a linear combination of the independent variables and their respective weights and bias (or the intercept) term for each neuron. The neural network equation looks like this:
- $Z = Bias + W_1X_1 + W_2X_2 + \ldots + W_nX_n$
- Z is the symbol for denotation of ANN.
- W is, are the weights or the beta coefficients.(weights are the info used by the ANN to solve the problem)
- X is, are the independent variables or the inputs, and
- Bias or intercept = W_0 (In the cases of weighted sum is 0, bias is added to make the output not zero or to scale up the system response. Bias has the weight and input always = 0.)

Neuron Calculation



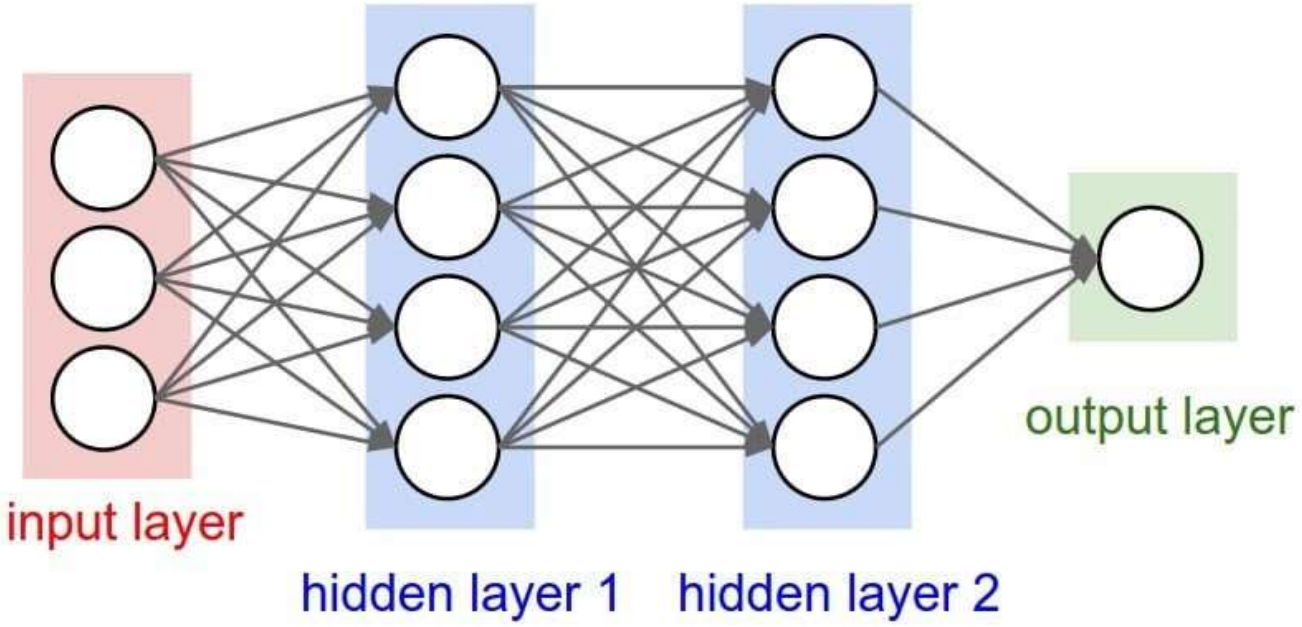
Step1: y=w1.x1+w2.x2+w3.x3…wn.xn+bias $\sum_{i=1}^{n} wi.xi$

Step 2: z= $\int(y)$

Activation function: if value is less than the threshold value than the neuron will not activated. If greater than the neuron will activated.

Multiple Layer Neural Network

- ANN comprises of multiple hidden layer to train on deeper level.



Multiple Layer Neural Network

- A neural network consists of three layers.
- **Input Layer:** The first layer is the input layer. It contains the input neurons that send information to the hidden layer.
- **Hidden Layer:** The hidden layer performs the computations on input data and transfers the output to the output layer. It includes weight, activation function, cost function.
- **Output Layer:** Final processing output.
- The connection between neurons is known as weight, which is the numerical values. The weight between neurons determines the learning ability of the neural network. During the learning of artificial neural networks, weight between the neuron changes.

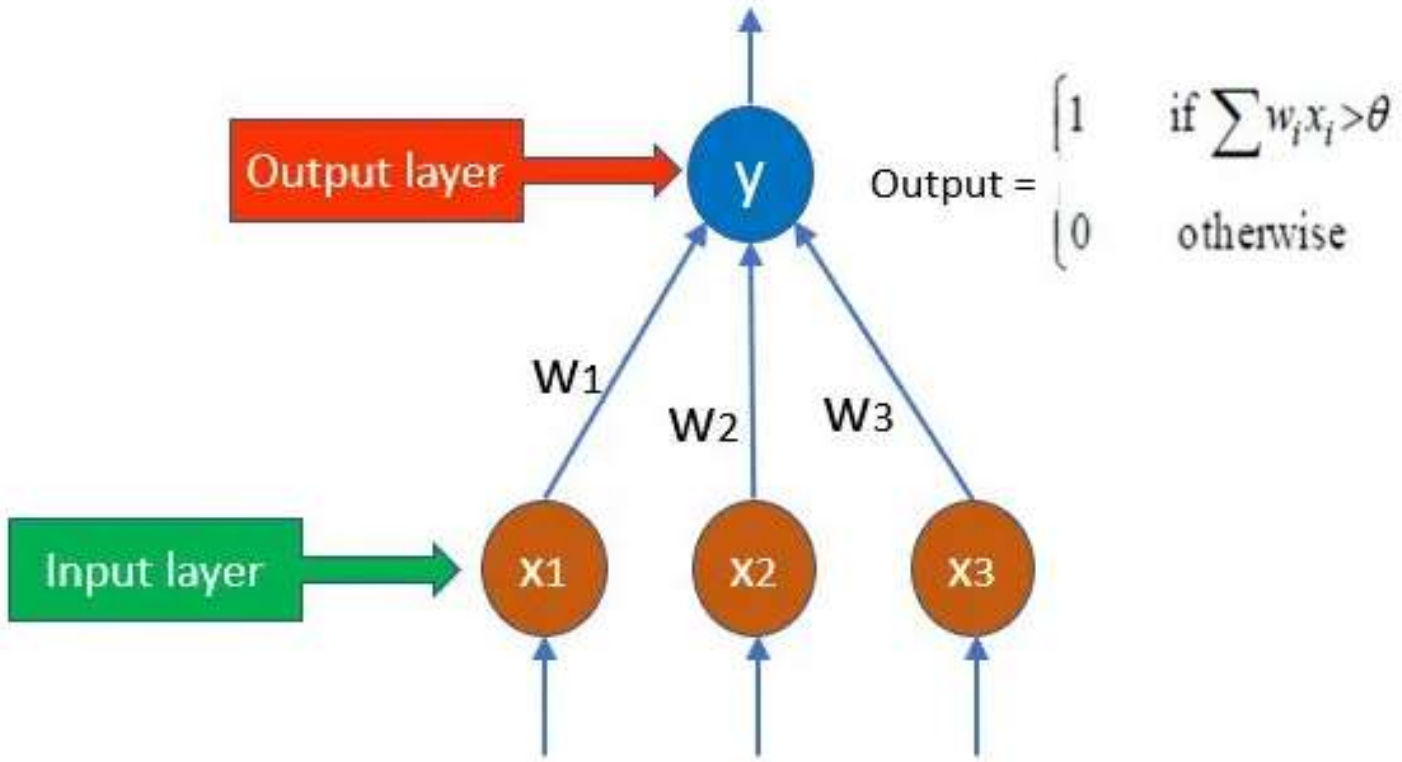
Training Algorithm of ANN

- **Gradient Decent Algorithm:** Simplest training algorithm used in case of supervised training model. In case the actual output is different from the target output , the difference or error is find out . The gradient decent changes the error in such a manner to minimize this mistake.
- **Back Propagation:** It is the extension of the gradient based learning. Here, after finding the error, the error will back-propagate backward to the input layer via hidden layers. It is used in the case of multilayer NN.

Neural Network Architecture Type

1. **Single Layer Perceptron Model**
2. **Radial Basis Function Neural Network**
3. **Multi-Layer Perceptron Neural Network**
4. **Recurrent Neural Network**
5. **Hopfield Neural Network**
6. **Boltzman Machine Neural Network**

Single Layer Perceptron



Single Layer Perceptron

- Perceptron model, proposed by Minsky-Papert is one of the simplest and oldest models of Neuron. It is the smallest unit of neural network that does certain computations to detect features or business intelligence in the input data. It accepts weighted inputs, and apply the activation function to obtain the output as the final result. Perceptron is also known as TLU(threshold logic unit)
- Perceptron is a supervised learning algorithm that classifies the data into two categories, thus it is a binary classifier. A perceptron separates the input space into two categories by a hyperplane represented by the following equation:
- **Advantages of Perceptron:**

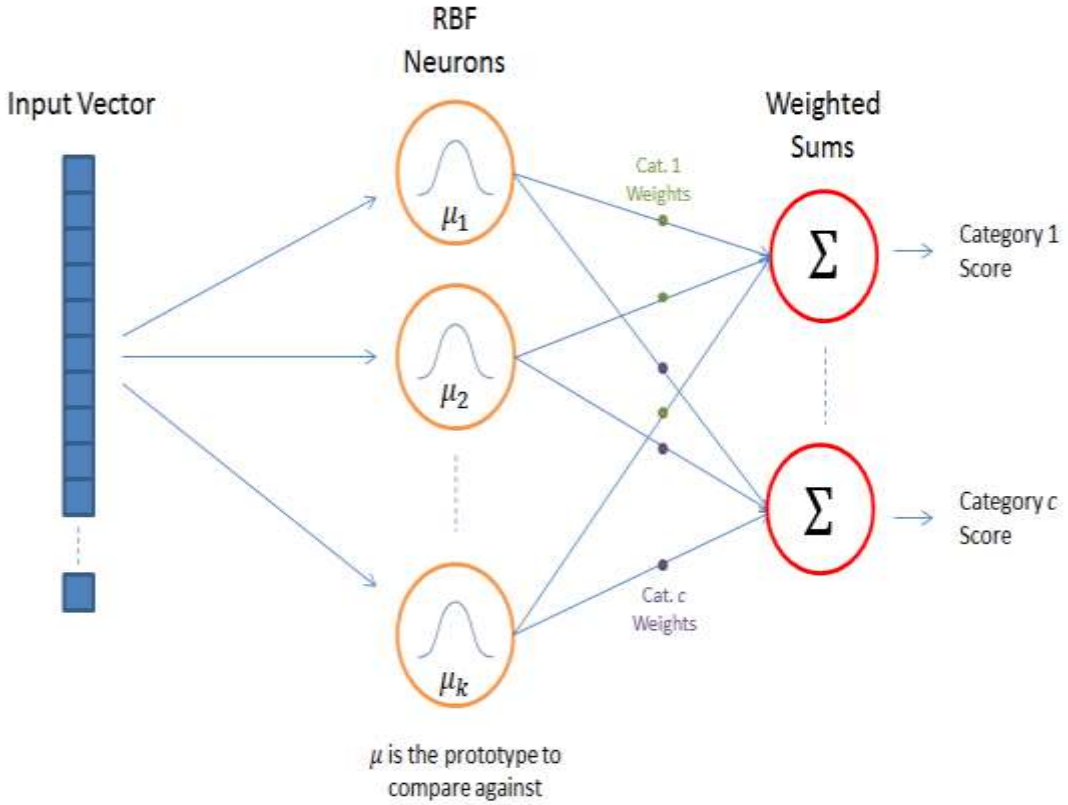
Perceptrons can implement Logic Gates like AND, OR, or NAND.

- **Disadvantages of Perceptron:**

Perceptrons can only learn linearly separable problems such as boolean AND problem. For non-linear problems such as the boolean XOR problem, it does not work.

Radial Basis Function Network

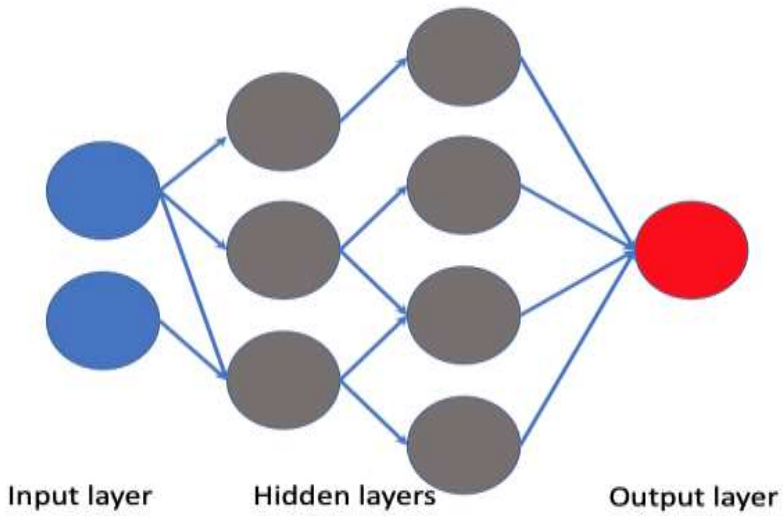
- A **Radial Basis Function** Network, or RBFN for short, is a form of neural network that relies on the integration of the Radial Basis Function and is specialized for tasks involving non-linear classification.



The RBFN approach is more intuitive than the MLP. The RBFN performs classification by measuring the input's similarity to examples from the training set. Each RBFN neuron stores a "prototype", which is just one of the examples from the training set. When we want to classify a new input, each neuron computes the Euclidean distance between the input and its prototype. Roughly speaking, if the input more closely resembles the class A prototypes than the class B prototypes, it is classified as class A.

Feed-Forward Neural Network Network

- Feed-forward neural networks allows signals to travel one approach only, from input to output. There is no feedback (loops) such as the output of some layer does not influence that same layer. Feed-forward networks tends to be simple networks that associates inputs with outputs. It can be used in pattern recognition. This type of organization is represented as bottom-up or top-down.
- A neural network can have several hidden layers, but as usual, one hidden layer is adequate. The wider the layer the higher the capacity of the network to identify designs.



Cost Function in Feed-Forward Neural Network

- The cost function is an important factor of a feed-forward neural network. Generally, minor adjustments to weights and biases have little effect on the categorized data points. Thus, to determine a method for improving performance by making minor adjustments to weights and biases using a smooth cost function.
- The mean square error cost function is defined as follows:

$$C(w,b) \equiv \frac{1}{2n} \sum_x \|y(x) - a\|^2.$$

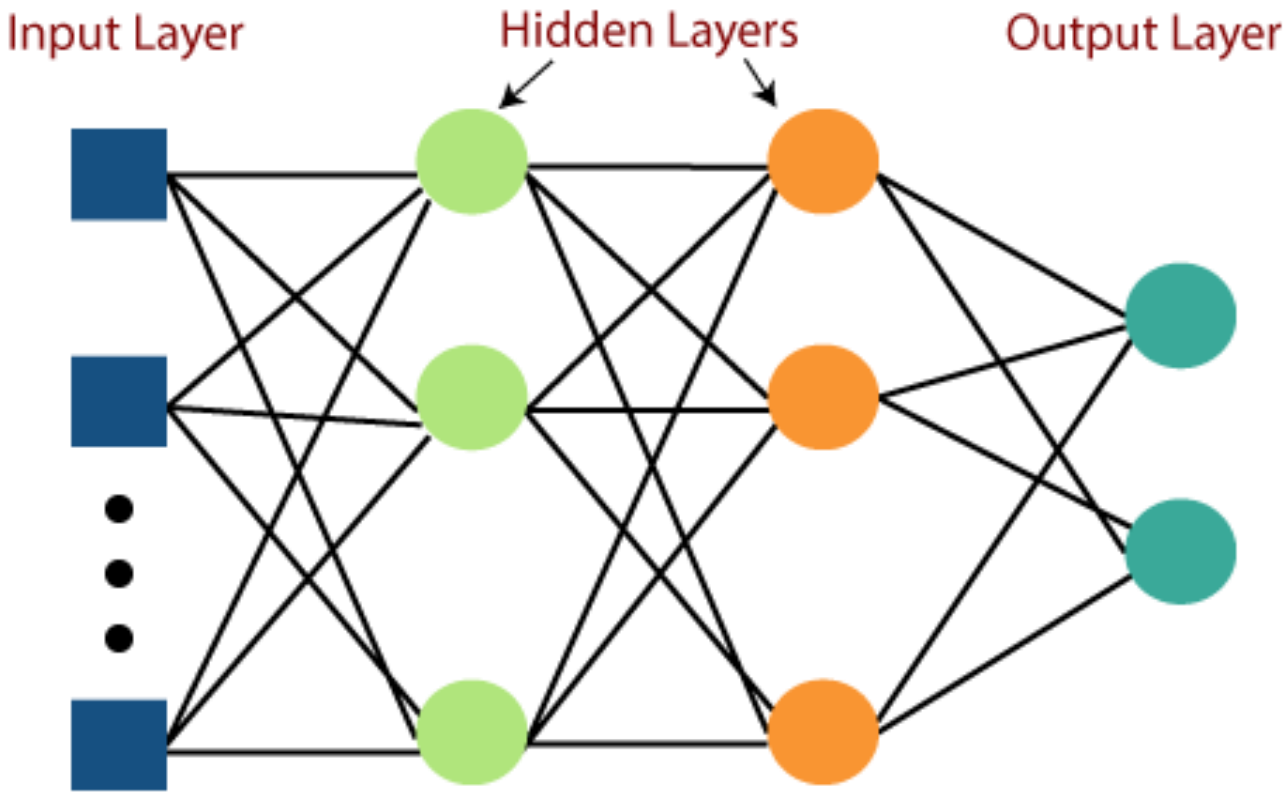
- w = weights collected in the network
- b = biases
- n = number of training inputs
- a = output vectors
- x = input
- ‖v‖ = usual length of vector v

Feed-Forward Neural Network Network

- **Advantages of Feed Forward Neural Networks**
- Less complex, easy to design & maintain
- Fast and speedy [One-way propagation]
- Highly responsive to noisy data
- **Disadvantages of Feed Forward Neural Networks:**
- Cannot be used for deep learning [due to absence of dense layers and back propagation]

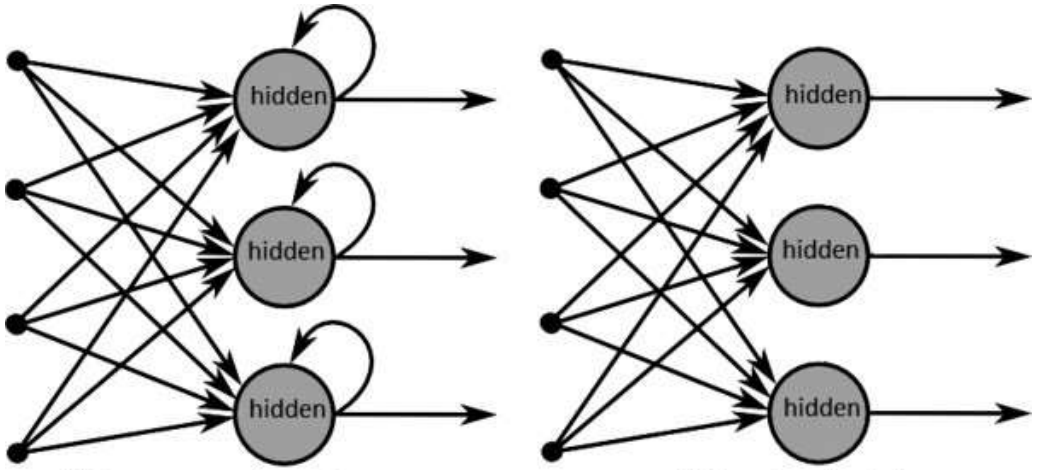
Multi-Layer Perceptron Network

- An entry point towards complex neural nets where input data travels through various layers of artificial neurons. Every single node is connected to all neurons in the next layer which makes it a fully connected neural network. Input and output layers are present having multiple hidden Layers i.e. at least three or more layers in total. It has a bi-directional propagation i.e. forward propagation and backward propagation.
- Inputs are multiplied with weights and fed to the activation function and in backpropagation, they are modified to reduce the loss. In simple words, weights are machine learnt values from Neural Networks. They self-adjust depending on the difference between predicted outputs vs training inputs. Nonlinear activation functions are used followed by softmax as an output layer activation function.
- **Advantages on Multi-Layer Perceptron**
- Used for deep learning [due to the presence of dense fully connected layers and back propagation]
- **Disadvantages on Multi-Layer Perceptron:**
- Comparatively complex to design and maintain
- Comparatively slow (depends on number of hidden layers)


Input Layer
Hidden Layers
Output Layer

Recurrent Neural Network

- **What is sequential data?**
- If there is a particular order in which related things follow each other, we call it as a **sequence.**
- **“i am a good boy”** and **“am i a good boy”** .
- Do you think both sentences mean the same? **NO! which means the position of words is very important! They are a sequence of words.**

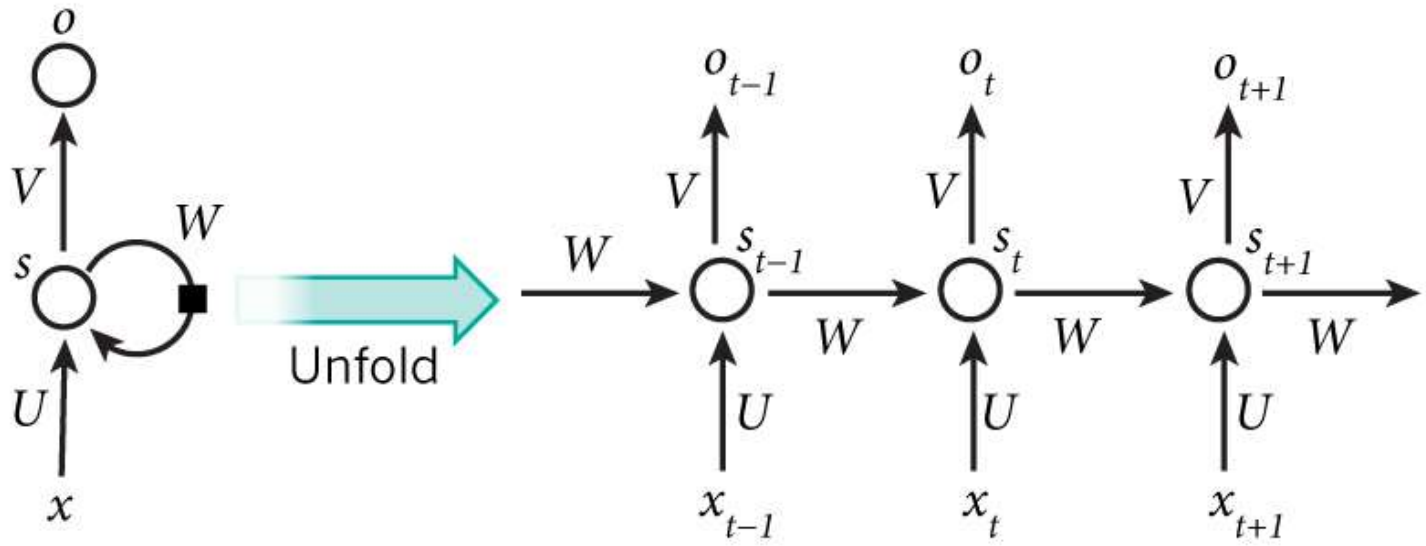


(a) Recurrent neural network

(b) Forward neural network

Recurrent Neural Network

- A recurrent neural network (RNN) is a type of artificial neural network which uses sequential data or time series data. These deep learning algorithms are commonly used for ordinal or temporal problems, such as language translation, natural language processing (nlp), speech recognition, and image captioning; they are incorporated into popular applications such as Siri, voice search, and Google Translate. Like feedforward and convolutional neural networks (CNNs), recurrent neural networks utilize training data to learn.
- They are distinguished by their “memory” as they take information from prior inputs to influence the current input and output. While traditional deep neural networks assume that inputs and outputs are independent of each other, the output of recurrent neural networks depend on the prior elements within the sequence. While future events would also be helpful in determining the output of a given sequence, unidirectional recurrent neural networks cannot account for these events in their predictions.



Here, x_t: input at time t, s_t: hidden state at time t, and O_t: output at time t

The Green Box represents a Neural Network. The arrows indicate memory or simply feedback to the next input.

The first figure shows the RNN. The Second figure shows the same RNN unrolled in time. **Applications:**

Generating Text

Machine Translation

Speech Recognition

Generating image description

Process of RNN

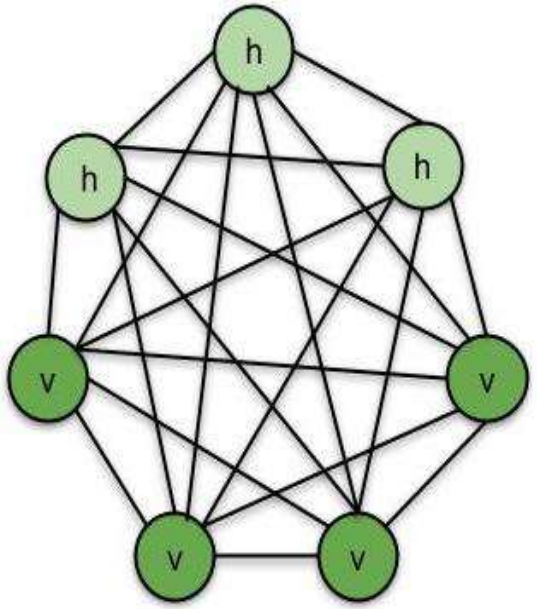
- The tree set of parameter (U, V, and W) are used to apply linear transformation over their respective inputs
- Parameter U transformation the input x_t to the state s_t
- Parameter W transforms the previous state s_{t-1} to the current state s_t
- And, parameter V maps the computed internal state s_t to the output O_t
- **Formula to calculate current state:**
- $\mathbf{h_t = f(h_{t-1}, x_t)}$
- Here, h_t is the current state, h_{t-1} is the previous state and x_t is the current input
- The equation applying after activation function (tanh) is:
- $\mathbf{h_t = \tanh(w_{hh} h_{t-1} + w_{xh} x_t)}$
- Here, whh : weight at recurrent neuron, Wxh : weight at input neuron
- After calculating the final state, we can then produce the output
- The output state can be calculated as:
- $O_t = W_{hy}\, h_t$
- Here, O_t is the output state, why: weight at output layer, h_t: current state

Hopfield Neural Network

- **It is a form of Recurrent Artificial Neural Network, invented by John Hopfield.** It is a type of ANN.
- Serve as content-addressable memory system with binary threshold units. Binary neurons with discrete time, updated one at a time
- $Vj(t+1)=\{1,0,\text{ if }\ \Sigma kTjkVk(t)+Ij>0$ otherwise
- Graded neurons with continuous time
- $dxj/dt=-xj/\tau+\Sigma kTjkg(xk)+Ij$.
- Here,Vj denotes activity of the j-th neuron.
- xj is the mean internal potential of the neuron.
- Ij is direct input (e.g., sensory input or bias current) to the neuron.
- Tjk is the strength of synaptic input from neuron k to neuron j .
- g is a monotone function that converts internal potential into firing rate output of the neuron, i.e., $Vj=g(xj)$.

Boltzmann Network

- **Boltzmann Machines** is an unsupervised DL model in which every node is connected to every other node. That is, unlike the ANNs, CNNs, RNNs and SOMs, the Boltzmann Machines are **undirected** (or the connections are bidirectional). Boltzmann Machine is not a deterministic DL model but a **stochastic** or **generative** DL model. It is rather a representation of a certain system. There are two types of nodes in the Boltzmann Machine — **Visible nodes** — those nodes which we can and do measure, and the **Hidden nodes** – those nodes which we cannot or do not measure. Although the node types are different, the Boltzmann machine considers them as the same and everything works as one single system. The training data is fed into the Boltzmann Machine and the weights of the system are adjusted accordingly. Boltzmann machines help us understand abnormalities by learning about the working of the system in normal conditions.



v - visible nodes, h - hidden nodes

Activation Function

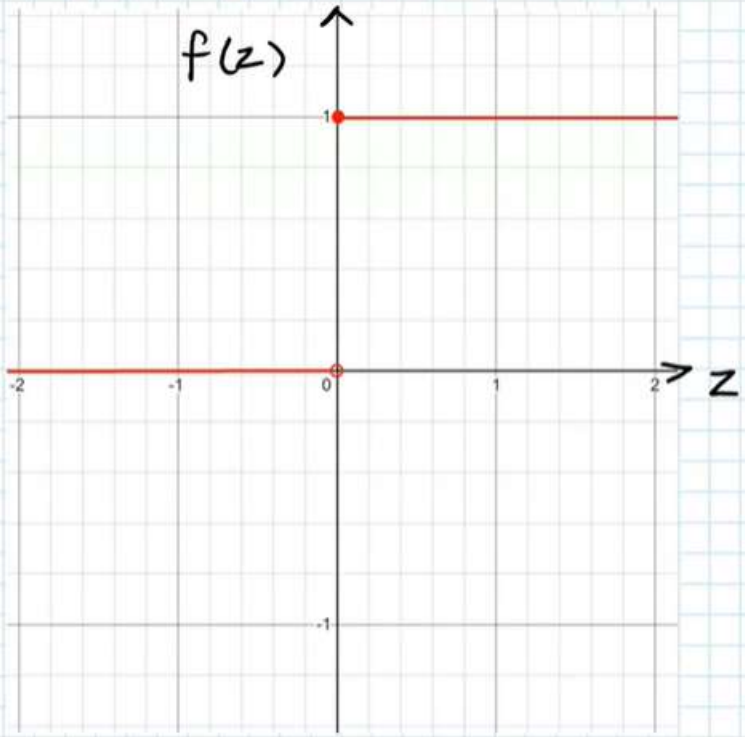
- Activation function decides, whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to **introduce non-linearity** into the output of a neuron.
- **Explanation :-** We know, neural network has neurons that work in correspondence of *weight*, *bias* and their respective activation function. In a neural network, we would update the weights and biases of the neurons on the basis of the error at the output. This process is known as *back-propagation*. Activation functions make the back-propagation possible since the gradients are supplied along with the error to update the weights and biases.
- **Why do we need Non-linear activation functions :-** A neural network without an activation function is essentially just a linear regression model. The activation function does the non-linear transformation to the input making it capable to learn and perform more complex tasks.
- There are various types of activation functions.

Step Function

Commonly Used Activation Functions

1. Step function

$$f(z) = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases}$$



Range = Possible Output

$\{0, 1\}$

Used in: Hidden Layer, Output Layer for Classification

Signum Function

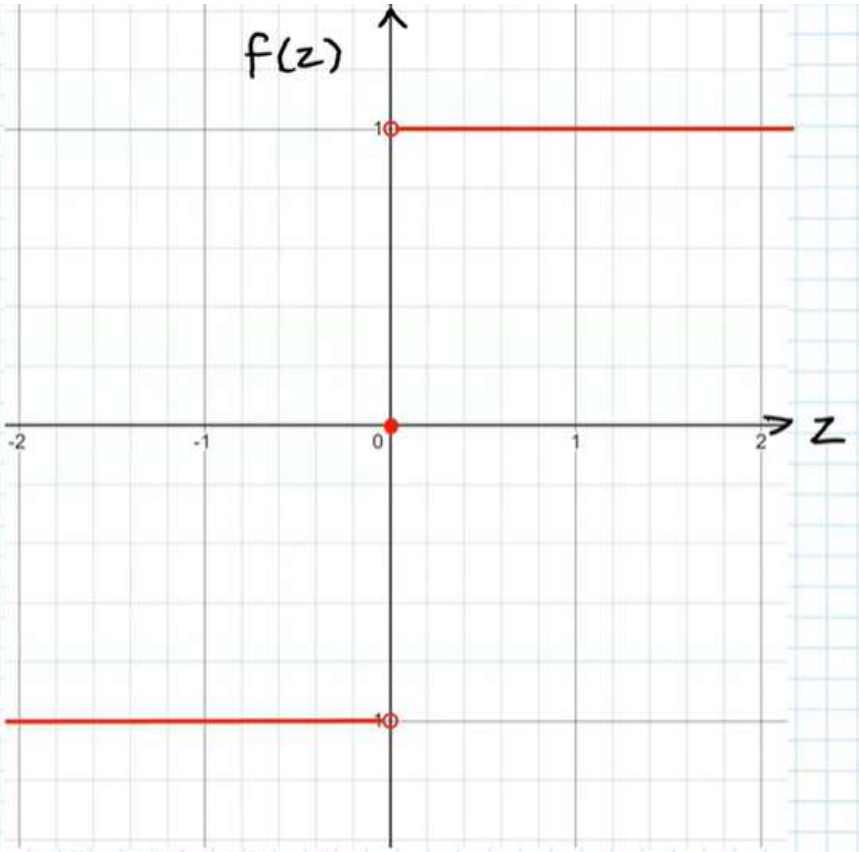
Commonly Used Activation Functions

2. Signum (sgn or sign) function

$$f(z) = \begin{cases} -1 & z < 0 \\ 1 & z > 0 \end{cases}$$

Range = Possible Outputs

$\{-1, 1\}$



Used in: Hidden Layer, Output Layer for Classification

Linear Function

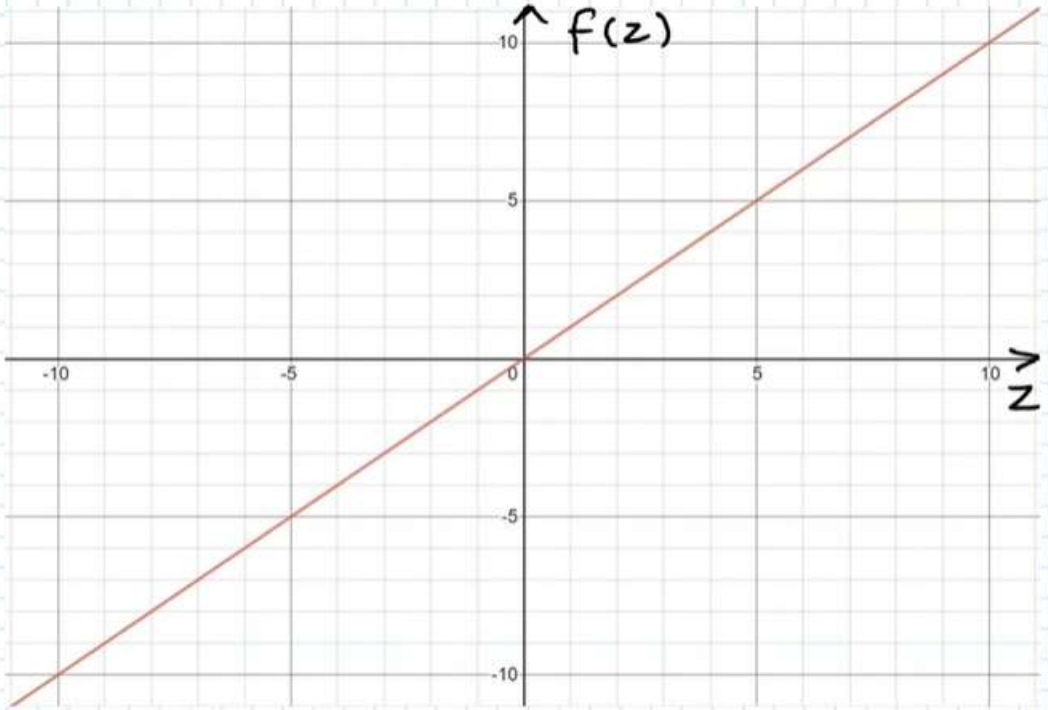
Commonly Used Activation Functions

3. Linear Function

$f(z) = z$

Range = Possible Outputs

$(-\infty, \infty)$



Used in: Hidden Layer, Output Layer for Regression

ReLU Function

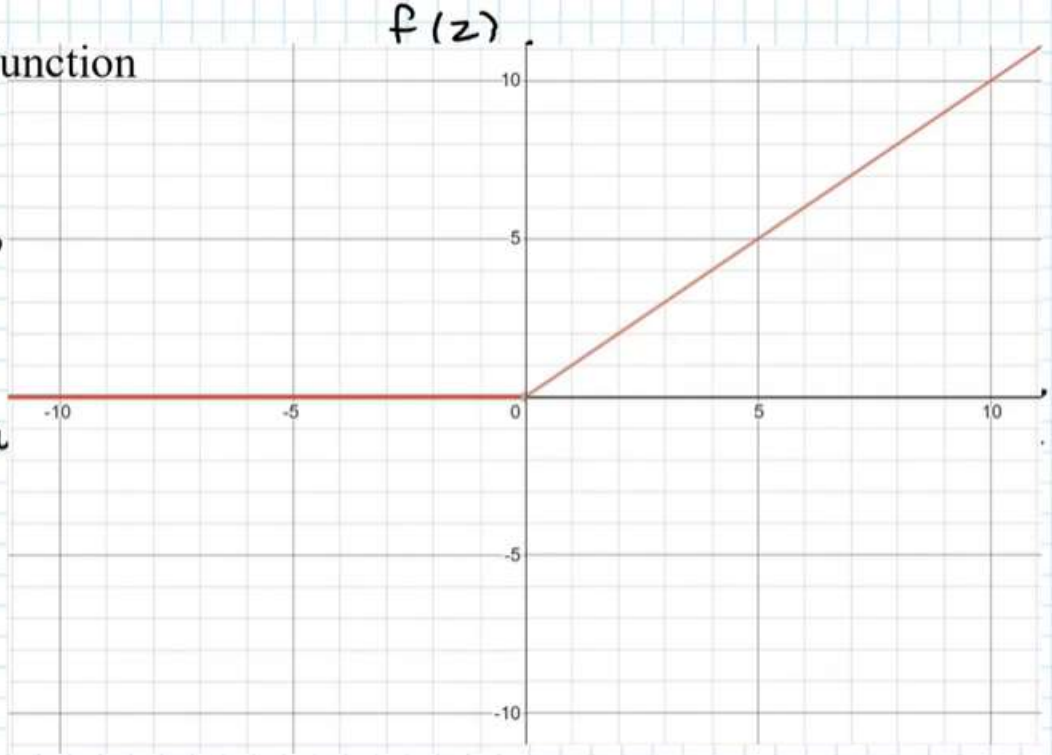
Commonly Used Activation Functions

4a. Rectified Linear Unit (ReLU) Function

$$f(z) = \begin{cases} 0 & z < 0 \\ z & z \geq 0 \end{cases}$$

Range = Possible Outpu

$[0, \infty)$



Used in: Hidden Layer, Output Layer for Regression (only positive output)

Leaky ReLU Function

Commonly Used Activation Functions

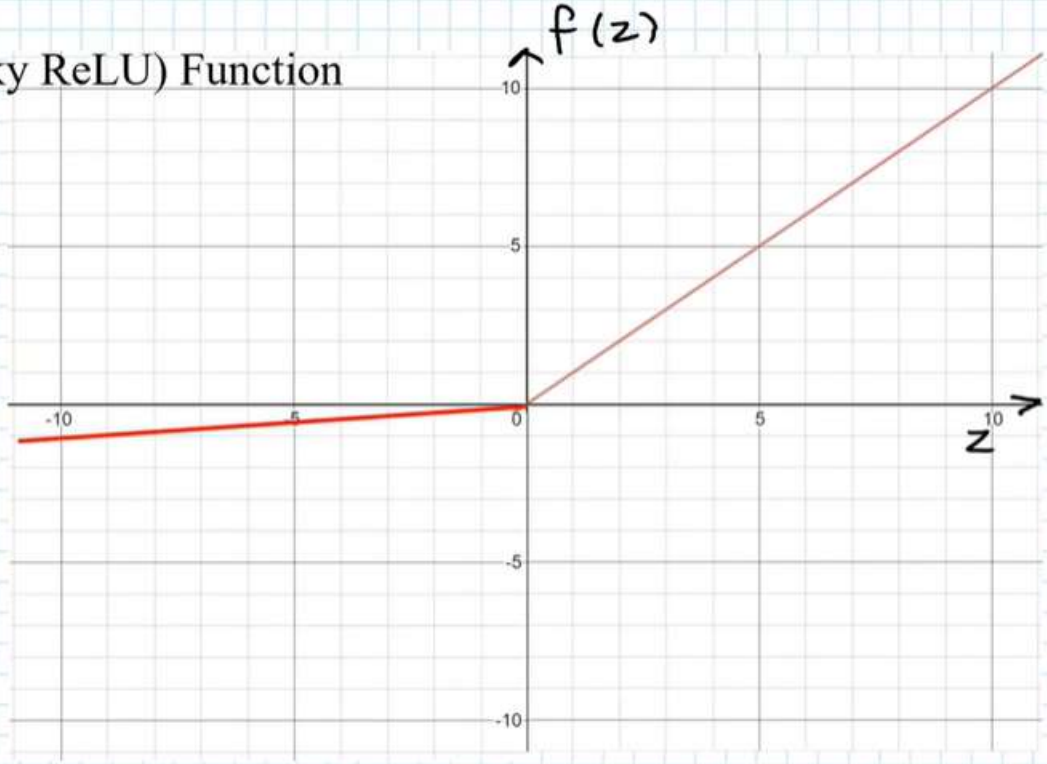
4b. Leaky Rectified Linear Unit (Leaky ReLU) Function

$$f(z) = \begin{cases} az & z < 0 \\ z & z \geq 0 \end{cases}$$

a is a small + number

Range = Possible outputs

$(-\infty, \infty)$



Used in: Hidden Layer

Hyperbolic Tangent Function

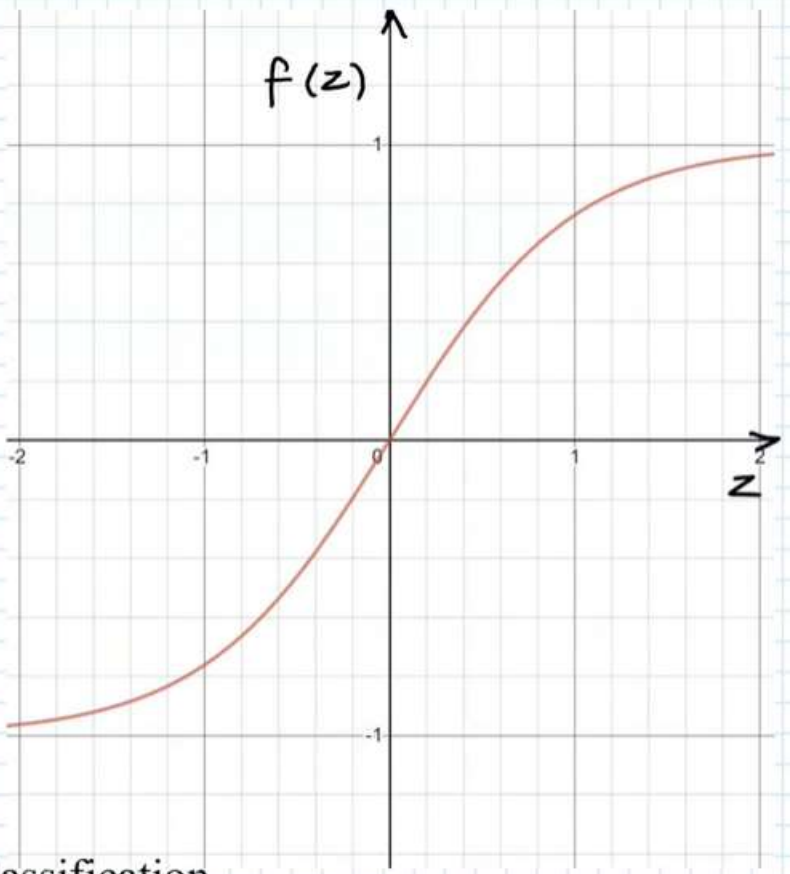
Commonly Used Activation Functions

5. Hyperbolic tangent; tanh(z)

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

Range = Possible Outputs

$(-1, 1)$



Used in: Hidden Layer, Output Layer for Classification

Sigmoid Function

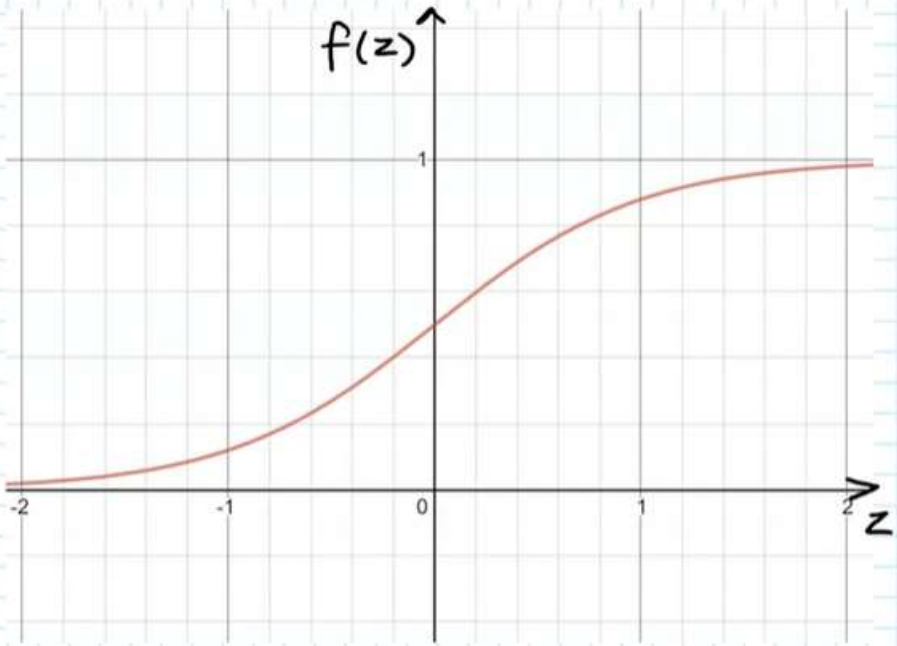
Commonly Used Activation Functions

6a. Sigmoid Function

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

Range = Possible Outputs

(0, 1)



Used in: Hidden Layer, Output Layer for Classification

Softmax Function

Commonly Used Activation Functions

6b. Softmax Function

$$z_1, z_2, \ldots, z_n$$

$$f(z_1) = \frac{e^{z_1}}{e^{z_1} + e^{z_2} + \cdots + e^{z_n}}$$

$$f(z_i) = \frac{e^{z_i}}{e^{z_1} + e^{z_2} + \cdots + e^{z_n}} = \frac{e^{z_i}}{\sum_{k=1}^{n} e^{z_k}}$$

Used in: Output Layer for Multiclass Classification

Backpropagation

- **Backpropagation** is the essence of neural network training. It is the method of fine-tuning the weights of a neural network based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows you to reduce error rates and make the model reliable by increasing its generalization.
- Backpropagation in neural network is a short form for “backward propagation of errors.” It is a standard method of training artificial neural networks. This method helps calculate the gradient of a loss function with respect to all the weights in the network.
- **Most prominent advantages of Backpropagation are:**
- Backpropagation is fast, simple and easy to program
- It has no parameters to tune apart from the numbers of input
- It is a flexible method as it does not require prior knowledge about the network
- It is a standard method that generally works well
- It does not need any special mention of the features of the function to be learned.

Backpropagation Algorithm

- **Input:**

D, a dataset consisting of the training tuples and their associated target values.

L, the learning rate.

Network, a multi-layer feed-forward network.

- **Output:**

A trained neural network.

- **Method:**

1. Initialize all the weights and biases in the network.
2. **While (training condition is not satisfied)**
3. {
4. **For each training tuple X in D**
5. { // propogate the input forward:

- **For each input layer unit j:**
- {
- Oj=Ij; # *output of input unit is its actual value*
- }
- **For each hidden or output layer unit j:**
- {

$$I_j = \sum w_{ij}.o_i + \theta_j$$

Where,

$w_{ij} -> is\ the$ weight from unit i in the previous layer to unit j.

$o_i -> output$ of unit i from previous unit.

$\theta_j -> is$ the bias of the current unit.

- Oj = $1/1+e^{-Ij}$
- }

 # Backpropagate the errors

- **For each unit j in the output layer:**
- {

$$Error_j = o_j(1-o_j)(T_j - o_j)$$

- }

- **For each unit j in the hidden layer**
- {

  $Error_j = O_j(1-o_j)\sum Error_k w_{jk}$

  Where,

  $w_{jk} ->$ is the weight of the connection from unit j to unit k.

  $Error_k ->$ is the error of unit k.

- }
- **For each weight wij in the network:**
- {

  $\Delta w_{ij} = (l) Error_j O_i$ #weight increment

  $w_{ij} = w_{ij} + \Delta w_{ij}$ # weight updation

- **Where** Δw_{ij} is the change in the weight
- }

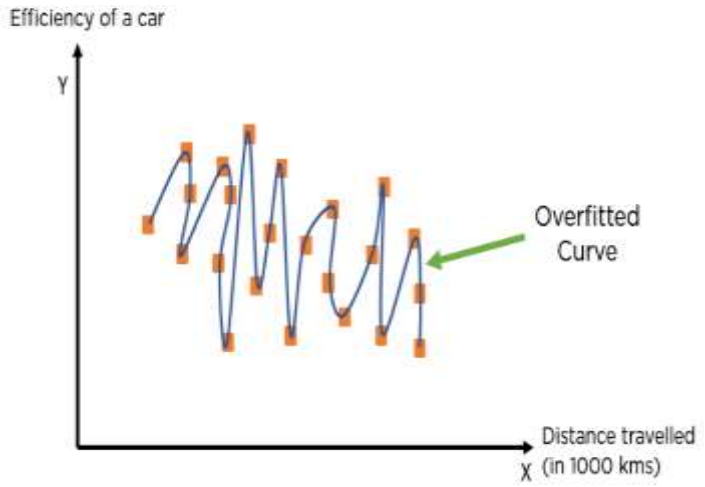
PRINCIPAL COMPONENT ANALYSIS (PCA)

Introduction

- Principal component analysis (PCA) is a technique that is useful for the compression and classification of data. The purpose is to reduce the dimensionality of a data set (sample) by finding a new set of variables, smaller than the original set of variables, that nonetheless retains most of the sample's information.
- PCA is used to reduce the problem of Overfitting.

Introduction to Overfitting and Underfitting

- **Overfitting:** When a model performs very well for training data but has poor performance with test data (new data), it is known as overfitting. In this case, the machine learning model learns the details and noise in the training data such that it negatively affects the performance of the model on test data. Overfitting can happen due to low bias and high variance.
- **Reasons for Overfitting**
- Data used for training is not cleaned and contains noise (garbage values) in it
- The model has a high variance
- The size of the training dataset used is not
- The model is too complex

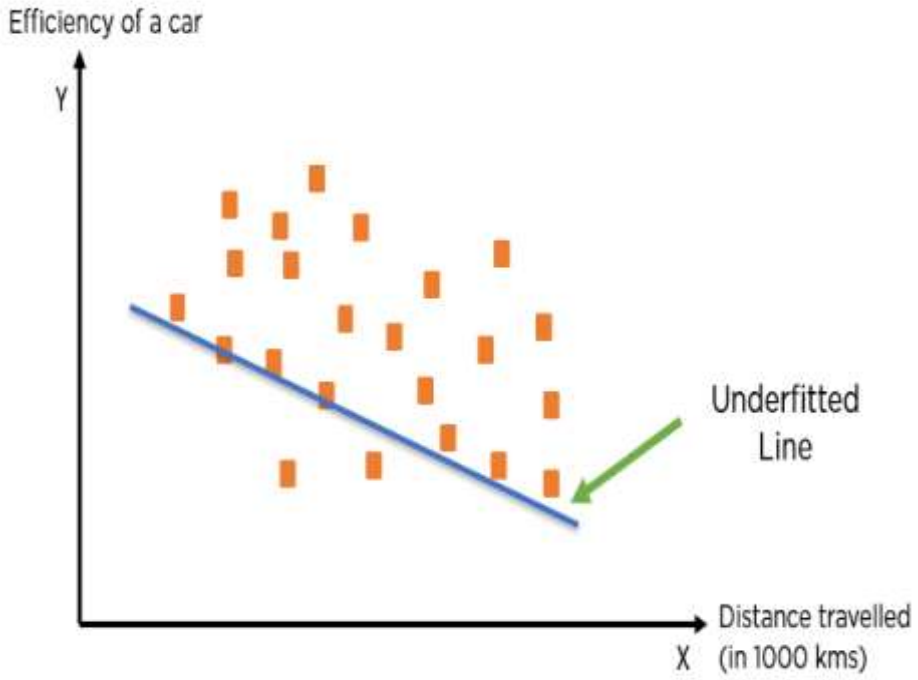


Introduction to Overfitting and Underfitting

- **Underfitting:** When a model has not learned the patterns in the training data well and is unable to generalize well on the new data, it is known as underfitting. An underfit model has poor performance on the training data and will result in unreliable predictions. Underfitting occurs due to high bias and low variance.

Reasons for Underfitting

- Data used for training is not cleaned and contains noise (garbage values) in it.
- The model has a high bias.
- The size of the training dataset used is not enough.
- The model is too simple.



Steps Included in PCA

- Step 1: Dataset Collection
- Step 2: Mean Calculation
- Step 3: Computation of Co-Variance Matrix
- Step 4: Eigen value, Eigen vector, and normalized Eigen vectors
- Step 5: Derive new Dataset.

Solved Numerical on PCA

Principal component analysis numerical

- Given data = { 2, 3, 4, 5, 6, 7 ; 1, 5, 3, 6, 7, 8 }.

Compute the principal component using PCA Algorithm.

- Consider the two dimensional patterns (2, 1), (3, 5), (4, 3), (5, 6), (6, 7), (7, 8).

Compute the principal component using PCA Algorithm.

Solved Numerical on PCA

Principal component analysis numerical

- **Step-01:** Get data.
- **Step-02:** Compute the mean vector (μ).
- **Step-03:** Subtract mean from the given data.
- **Step-04:** Calculate the covariance matrix.
- **Step-05:** Calculate the eigen vectors and eigen values of the covariance matrix.
- **Step-06:** Choosing components and forming a feature vector.
- **Step-07:** Deriving the new data set.

Solved Numerical on PCA

Principal component analysis numerical

- Compute the principal component of following data-
- CLASS 1
- X = 2 , 3 , 4
- Y = 1 , 5 , 3
- CLASS 2
- X = 5 , 6 , 7
- Y = 6 , 7 , 8

Solved Numerical on PCA

Principal component analysis numerical

- The given feature vectors are-
- $x_1 = (2, 1)$
- $x_2 = (3, 5)$
- $x_3 = (4, 3)$
- $x_4 = (5, 6)$
- $x_5 = (6, 7)$
- $x_6 = (7, 8)$

Solved Numerical on PCA

Principal component analysis numerical

- **Step-02:**

- Calculate the mean vector (μ).

- Mean vector (μ)
- = ((2 + 3 + 4 + 5 + 6 + 7) / 6, (1 + 5 + 3 + 6 + 7 + 8) / 6)
- = (4.5, 5)

Solved Numerical on PCA

Principal component analysis numerical

- Subtract mean vector (μ) from the given feature vectors.
- $x_1 - \mu = (2 - 4.5, 1 - 5) = (-2.5, -4)$
- $x_2 - \mu = (3 - 4.5, 5 - 5) = (-1.5, 0)$
- $x_3 - \mu = (4 - 4.5, 3 - 5) = (-0.5, -2)$
- $x_4 - \mu = (5 - 4.5, 6 - 5) = (0.5, 1)$
- $x_5 - \mu = (6 - 4.5, 7 - 5) = (1.5, 2)$
- $x_6 - \mu = (7 - 4.5, 8 - 5) = (2.5, 3)$

$$\begin{bmatrix} -2.5 \\ -4 \end{bmatrix} \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} \begin{bmatrix} 2.5 \\ 3 \end{bmatrix}$$

Solved Numerical on PCA

Principal component analysis numerical

- Calculate the covariance matrix.
- Covariance matrix is given by-

$$\text{Covariance Matrix} = \frac{\Sigma (x_i - \mu)(x_i - \mu)^t}{n}$$

Solved Numerical on PCA

$$m_1 = (x_1 - \mu)(x_1 - \mu)^t = \begin{bmatrix} -2.5 \\ -4 \end{bmatrix} \begin{bmatrix} -2.5 & -4 \end{bmatrix} = \begin{bmatrix} 6.25 & 10 \\ 10 & 16 \end{bmatrix}$$

$$m_2 = (x_2 - \mu)(x_2 - \mu)^t = \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} \begin{bmatrix} -1.5 & 0 \end{bmatrix} = \begin{bmatrix} 2.25 & 0 \\ 0 & 0 \end{bmatrix}$$

$$m_3 = (x_3 - \mu)(x_3 - \mu)^t = \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} \begin{bmatrix} -0.5 & -2 \end{bmatrix} = \begin{bmatrix} 0.25 & 1 \\ 1 & 4 \end{bmatrix}$$

$$m_4 = (x_4 - \mu)(x_4 - \mu)^t = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} \begin{bmatrix} 0.5 & 1 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

$$m_5 = (x_5 - \mu)(x_5 - \mu)^t = \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} \begin{bmatrix} 1.5 & 2 \end{bmatrix} = \begin{bmatrix} 2.25 & 3 \\ 3 & 4 \end{bmatrix}$$

$$m_6 = (x_6 - \mu)(x_6 - \mu)^t = \begin{bmatrix} 2.5 \\ 3 \end{bmatrix} \begin{bmatrix} 2.5 & 3 \end{bmatrix} = \begin{bmatrix} 6.25 & 7.5 \\ 7.5 & 9 \end{bmatrix}$$

Solved Numerical on PCA

Principal component analysis numerical

- Now,
- Covariance matrix
- $= (m_1 + m_2 + m_3 + m_4 + m_5 + m_6) / 6$
- On adding the above matrices and dividing by 6, we get-

$$\text{Covariance Matrix} = \frac{1}{6}\begin{bmatrix} 17.5 & 22 \\ 22 & 34 \end{bmatrix}$$

$$\text{Covariance Matrix} = \begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix}$$

Solved Numerical on PCA

Principal component analysis numerical

- Calculate the eigen values and eigen vectors of the covariance matrix.
- λ is an eigen value for a matrix M if it is a solution of the characteristic equation $|M - \lambda I| = 0$.
- So, we have-

$$\begin{vmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{vmatrix} - \begin{vmatrix} \lambda & 0 \\ 0 & \lambda \end{vmatrix} = 0$$

$$\begin{vmatrix} 2.92 - \lambda & 3.67 \\ 3.67 & 5.67 - \lambda \end{vmatrix} = 0$$

Solved Numerical on PCA

- From here,
- $(2.92 - \lambda)(5.67 - \lambda) - (3.67 \times 3.67) = 0$
- $16.56 - 2.92\lambda - 5.67\lambda + \lambda^2 - 13.47 = 0$
- $\lambda^2 - 8.59\lambda + 3.09 = 0$
- Solving this quadratic equation, we get $\lambda = 8.22, 0.38$
- Thus, two eigen values are $\lambda_1 = 8.22$ and $\lambda_2 = 0.38$.
- Clearly, the second eigen value is very small compared to the first eigen value.
- So, the second eigen vector can be left out.
- Eigen vector corresponding to the greatest eigen value is the principal component for the given data set.
- So. we find the eigen vector corresponding to eigen value λ_1.
- We use the following equation to find the eigen vector-
- $MX = \lambda X$
- where-
- M = Covariance Matrix
- X = Eigen vector
- λ = Eigen value

Solved Numerical on PCA

Substituting the values in the above equation, we get-

$$\begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix} \begin{bmatrix} X1 \\ X2 \end{bmatrix} = 8.22 \begin{bmatrix} X1 \\ X2 \end{bmatrix}$$

- Solving these, we get-
- $2.92X_1 + 3.67X_2 = 8.22X_1$
- $3.67X_1 + 5.67X_2 = 8.22X_2$
- On simplification, we get-
- $5.3X_1 = 3.67X_2$ .....(1)
- $3.67X_1 = 2.55X_2$ .....(2)
- From (1) and (2), $\mathbf{X_1 = 0.69X_2}$
- From (2), the eigen vector is-

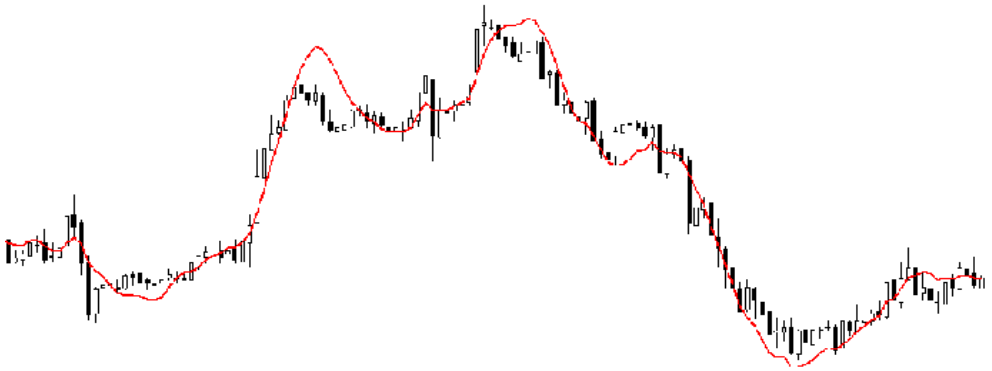
Eigen Vector : $\begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$

Principal Component : $\begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$

Regression Modeling

Introduction

- Regression analysis is a form of predictive modelling technique which investigates the relationship between a **dependent** (target) and **independent variable (s)** (predictor). This technique is used for forecasting, time series modelling and finding the casual effect analysis between the variables. For example, relationship between rash driving and number of road accidents by a driver is best studied through regression.

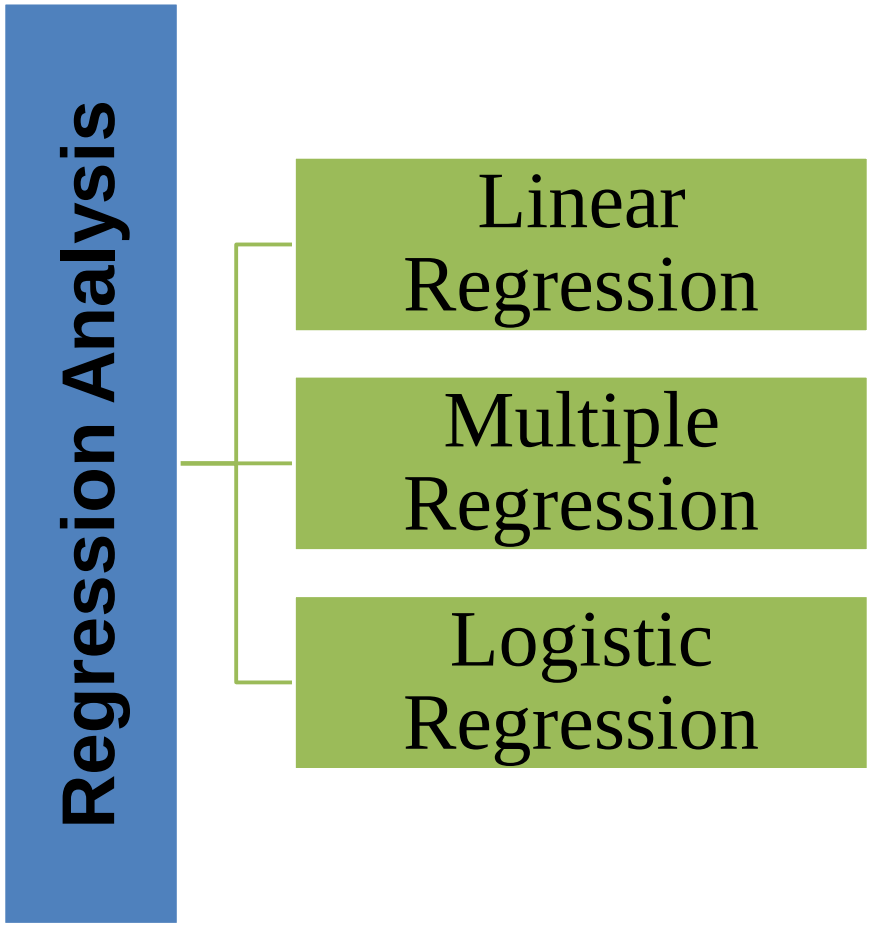


- Regression analysis is an important tool for modelling and analyzing data. Here, we fit a curve / line to the data points, in such a manner that the differences between the distances of data points from the curve or line is minimized. I'll explain this in more details in coming sections.

Need of Regression Analysis

- As mentioned above, regression analysis estimates the relationship between two or more variables. Let’s understand this with an easy example:
- Let’s say, you want to estimate growth in sales of a company based on current economic conditions. You have the recent company data which indicates that the growth in sales is around two and a half times the growth in the economy. Using this insight, we can predict future sales of the company based on current & past information.
- There are multiple benefits of using regression analysis. They are as follows:
- It indicates the **significant relationships** between dependent variable and independent variable.
- It indicates the **strength of impact** of multiple independent variables on a dependent variable.
- Regression analysis also allows us to compare the effects of variables measured on different scales, such as the effect of price changes and the number of promotional activities. These benefits help market researchers / data analysts / data scientists to eliminate and evaluate the best set of variables to be used for building predictive models.

Types of Regression

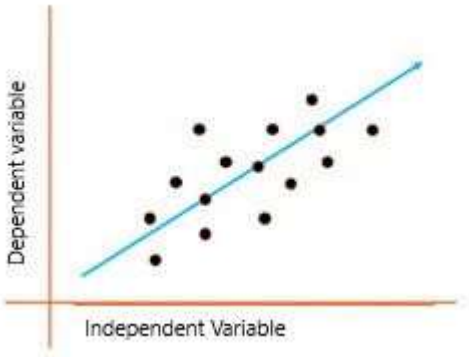


Types of Regression

- **Simple linear regression**
- One dependent variable (interval or ratio)
- One independent variable (interval or ratio or dichotomous)
- **Multiple linear regression**
- One dependent variable (interval or ratio)
- Two or more independent variables (interval or ratio or dichotomous)
- **Logistic regression**
- One dependent variable (binary)
- Two or more independent variable(s) (interval or ratio or dichotomous)
- **Ordinal regression**
- One dependent variable (ordinal)
- One or more independent variable(s) (nominal or dichotomous)
- **Multinomial regression**
- One dependent variable (nominal)
- One or more independent variable(s) (interval or ratio or dichotomous)
- **Discriminant analysis**
- One dependent variable (nominal)
- One or more independent variable(s) (interval or ratio)

Linear Regression

- Linear regression is a quiet and the simplest statistical regression method used for predictive analysis in machine learning. 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. If there is a single input variable **X**(dependent variable), such linear regression is called ***simple linear regression***.



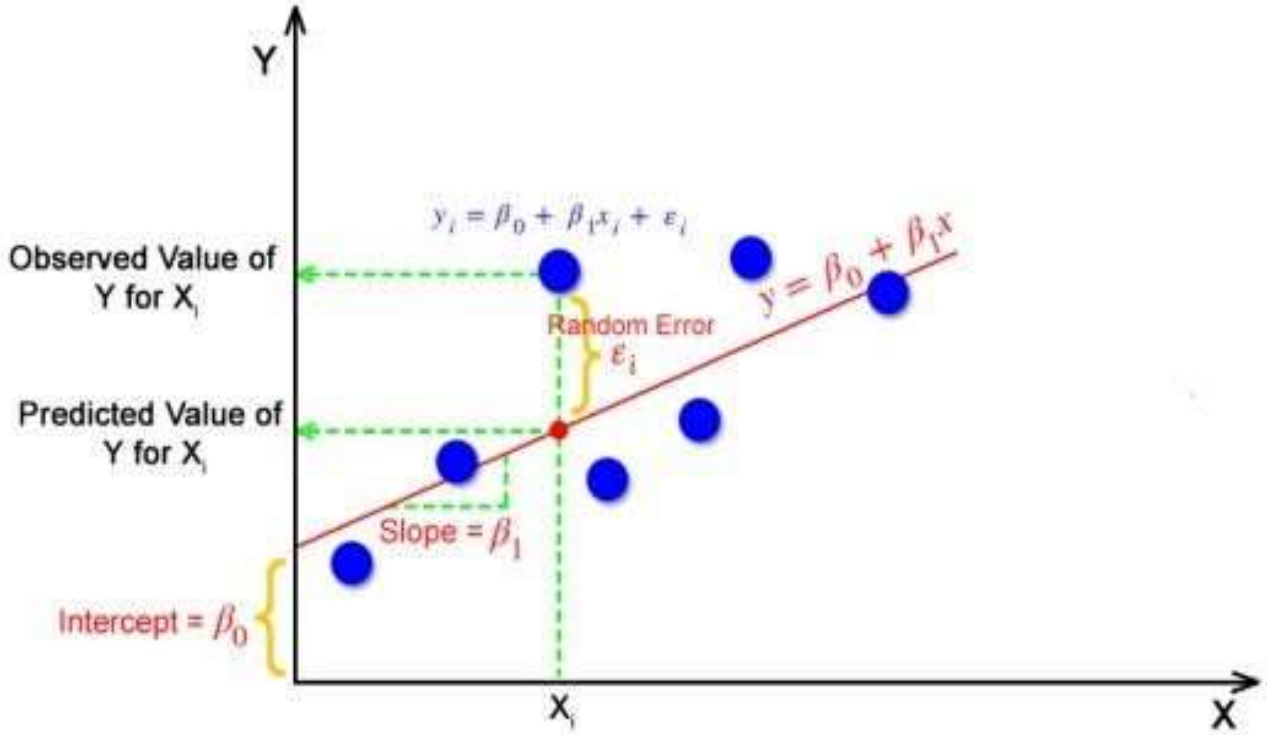
The above graph presents the linear relationship between the output(y) variable and predictor(X) variables. The blue line is referred to as the *best fit* straight line. Based on the given data points, we attempt to plot a line that fits the points the best.

Linear Regression

- Calculate best-fit line linear regression uses a traditional slope-intercept form which is given below,
- $\mathbf{Y_i = \beta_0 + \beta_1 X_i}$
- Where Y_i = Dependent variable to the given value of Independent variable.
- $\boldsymbol{\beta_0}$ = constant/Intercept the predicted value of y when x is 0.
- $\boldsymbol{\beta_1}$ = Slope or regression coefficient (how much we except y to change as x increase).
- $\mathbf{X_i}$ = Independent variable (The variable we expect influencing the dependent variable y).
- This algorithm explains the linear relationship between the dependent(output) variable y and the independent(predictor) variable X using a straight line.

➢ But how the linear regression finds out which is the best fit line?

The goal of the linear regression algorithm is to get the **best values for $\mathbf{B_0}$ and $\mathbf{B_1}$** 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.



- You can use the simple linear regression when you want to know:

1. How strong the relationship between two variables.
2. The value of dependent variable at a certain value of independent variable.

Assumptions Linear Regression

1. **Homogeneity of Variance:** The size of the error in our prediction doesn't change significantly across the value of independent variable.
2. **Independence of observations:** the observations in the dataset were collected using statistically valid sampling methods, and there is no hidden relationships among variables.
3. **Normality:** The data follows a normal distribution.

➢**Simple LR makes one additional assumption.**

1. The relationship between the independent and dependent variable is linear. The line of the best fit through the data point is a straight line rather than a curve.

Calculate R^2

- To check the goodness of fit, R^2 is calculated.
- R-Squarred value is the statistical measure of how close the data are to the fitted regression line.
- It determines that x and y are correlated or not. Large value shows the better model.
- For this calculate:
- Distance of actual-mean
- Distance of predicted-mean
- This is $R^{2=}$ $\dfrac{\sum (y_p - \bar{y})^2}{\sum (y - \bar{y})^2}$

Linear Regression Solved Numerical

Linear Regression Formula

As we know, linear regression shows the linear relationship between two variables. The equation of linear regression is similar to that of the slope formula. We have learned this formula before in earlier classes such as a linear equation in two variables. Linear Regression Formula is given by the equation

$Y = a + bX$

We will find the value of a and b by using the below formula

$$a = \frac{(\sum Y)(\sum X^2) - (\sum X)(\sum XY)}{n(\sum x^2) - (\sum x)^2}$$

$$b = \frac{n(\sum XY) - (\sum X)(\sum Y)}{n(\sum x^2) - (\sum x)^2}$$

Linear Regression Solved Numerical

Solved Examples

1. Find a linear regression equation for the following two sets of data:

x	2	4	6	8
y	3	7	5	10

Sol: To find the linear regression equation we need to find the value of Σx, Σy, Σx

2

2

and Σxy

Linear Regression Solved Numerical

Construct the table and find the value

x	**y**	**x^2**	**xy**
2	3	4	6
4	7	16	28
6	5	36	30
8	10	64	80
$\Sigma x = 20$	$\Sigma y = 25$	$\Sigma x^2 = 120$	$\Sigma xy = 144$

Linear Regression Solved Numerical

$$a = \frac{(\sum Y)(\sum X^2) - (\sum X)(\sum XY)}{n(\sum x^2) - (\sum x)^2}$$

Now put the values in the equation

$$a = \frac{25 \times 120 - 20 \times 144}{4 \times 120 - 400}$$

$$a = \frac{120}{80}$$

a=1.5

$$b = \frac{n(\sum XY) - (\sum X)(\sum Y)}{n(\sum x^2) - (\sum x)^2}$$

Put the values in the equation

$$b = \frac{4 \times 144 - 20 \times 25}{4 \times 120 - 400}$$

$$b = \frac{76}{80}$$

b=0.95

Hence we got the value of a = 1.5 and b = 0.95

Hence we got the value of a = 1.5 and b = 0.95

The linear equation is given by

Y = a + bx

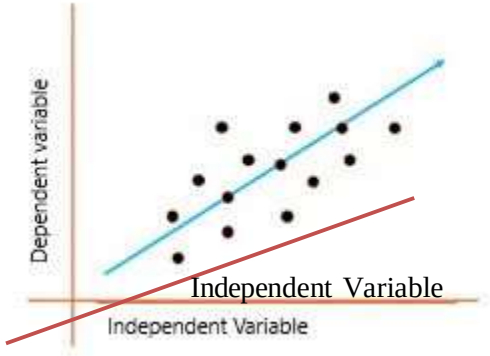
Now put the value of a and b in the equation

Hence equation of linear regression is y = 1.5 + 0.95x

x	y	yp
2	3	3.4
4	7	5.3
6	5	7.2
8	10	9.1
20/4=5	25/4=6.5	25

Multiple Linear Regression

- Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The goal of multiple linear regression is to model the linear relationship between the explanatory (independent) variables and response (dependent) variables.
-
- $y_i=\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+...+\beta_p x_{ip}+\epsilon$
- where, for i=n observations:
- yi=dependent variable
- xi=explanatory variables
- β0=y intercept (constant term)
- βp=slope coefficients for each explanatory variable
- ϵ=the model's error term (also known as the residuals)
- As the number of independent variable increases to 2 our graph become 3D. The added 3rd dimension represents other independent variable.



Multiple Linear Regression Numerical

Suppose we have the following dataset with one response variable *y* and two predictor variables X_1 and X_2:

y	X_1	X_2
140	60	22
155	62	25
159	67	24
179	70	20
192	71	15
200	72	14
212	75	14
215	78	11

Use the following steps to fit a multiple linear regression model to this dataset.

Multiple Linear Regression Numerical

Step 1: Calculate X_1^2, X_2^2, X_1y, X_2y and X_1X_2.

	y	X_1	X_2
	140	60	22
	155	62	25
	159	67	24
	179	70	20
	192	71	15
	200	72	14
	212	75	14
	215	78	11
Mean	181.5	69.375	18.125
Sum	1452	555	145

X_1^2	X_2^2	X_1y	X_2y	X_1X_2
3600	484	8400	3080	1320
3844	625	9610	3875	1550
4489	576	10653	3816	1608
4900	400	12530	3580	1400
5041	225	13632	2880	1065
5184	196	14400	2800	1008
5625	196	15900	2968	1050
6084	121	16770	2365	858
38767	2823	101895	25364	9859

Sum

Multiple Linear Regression Numerical

Step 2: Calculate Regression Sums.

Next, make the following regression sum calculations:

- $\Sigma x_1^2 = \Sigma X_1^2 - (\Sigma X_1)^2 / n = 38{,}767 - (555)^2 / 8 =$ **263.875**
- $\Sigma x_2^2 = \Sigma X_2^2 - (\Sigma X_2)^2 / n = 2{,}823 - (145)^2 / 8 =$ **194.875**
- $\Sigma x_1 y = \Sigma X_1 y - (\Sigma X_1 \Sigma y) / n = 101{,}895 - (555*1{,}452) / 8 =$ **1,162.5**
- $\Sigma x_2 y = \Sigma X_2 y - (\Sigma X_2 \Sigma y) / n = 25{,}364 - (145*1{,}452) / 8 =$ **-953.5**
- $\Sigma x_1 x_2 = \Sigma X_1 X_2 - (\Sigma X_1 \Sigma X_2) / n = 9{,}859 - (555*145) / 8 =$ **-200.375**

Multiple Linear Regression Numerical

	y	X_1	X_2
	140	60	22
	155	62	25
	159	67	24
	179	70	20
	192	71	15
	200	72	14
	212	75	14
	215	78	11
Mean	181.5	69.375	18.125
Sum	1452	555	145

X_1^2	X_2^2	X_1y	X_2y	X_1X_2
3600	484	8400	3080	1320
3844	625	9610	3875	1550
4489	576	10653	3816	1608
4900	400	12530	3580	1400
5041	225	13632	2880	1065
5184	196	14400	2800	1008
5625	196	15900	2968	1050
6084	121	16770	2365	858
Sum 38767	2823	101895	25364	9859

Reg Sums	263.875	194.875	1162.5	-953.5	-200.375

Multiple Linear Regression Numerical

	y	X_1	X_2
	140	60	22
	155	62	25
	159	67	24
	179	70	20
	192	71	15
	200	72	14
	212	75	14
	215	78	11
Mean	181.5	69.375	18.125
Sum	1452	555	145

X_1^2	X_2^2	X_1y	X_2y	X_1X_2
3600	484	8400	3080	1320
3844	625	9610	3875	1550
4489	576	10653	3816	1608
4900	400	12530	3580	1400
5041	225	13632	2880	1065
5184	196	14400	2800	1008
5625	196	15900	2968	1050
6084	121	16770	2365	858
Sum 38767	2823	101895	25364	9859

Reg Sums	263.875	194.875	1162.5	-953.5	-200.375

Multiple Linear Regression Numerical

Step 3: Calculate b_0, b_1, and b_2.

The formula to calculate b_1 is: $[(\Sigma x_2^2)(\Sigma x_1 y) - (\Sigma x_1 x_2)(\Sigma x_2 y)] / [(\Sigma x_1^2)(\Sigma x_2^2) - (\Sigma x_1 x_2)^2]$

Thus, $\mathbf{b_1} = [(194.875)(1162.5) - (-200.375)(-953.5)] / [(263.875)(194.875) - (-200.375)^2] = \mathbf{3.148}$

The formula to calculate b_2 is: $[(\Sigma x_1^2)(\Sigma x_2 y) - (\Sigma x_1 x_2)(\Sigma x_1 y)] / [(\Sigma x_1^2)(\Sigma x_2^2) - (\Sigma x_1 x_2)^2]$

Thus, $\mathbf{b_2} = [(263.875)(-953.5) - (-200.375)(1152.5)] / [(263.875)(194.875) - (-200.375)^2] = \mathbf{-1.656}$

The formula to calculate b_0 is: $\overline{y} - b_1\overline{X}_1 - b_2\overline{X}_2$

Thus, $\mathbf{b_0} = 181.5 - 3.148(69.375) - (-1.656)(18.125) = \mathbf{-6.867}$

Multiple Linear Regression Numerical

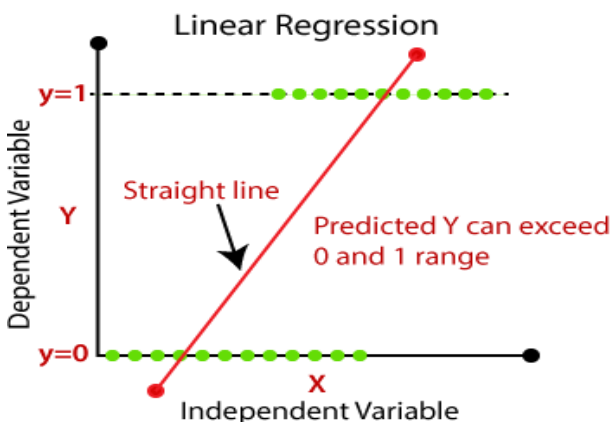
Step 5: Place b_0, b_1, and b_2 in the estimated linear regression equation.

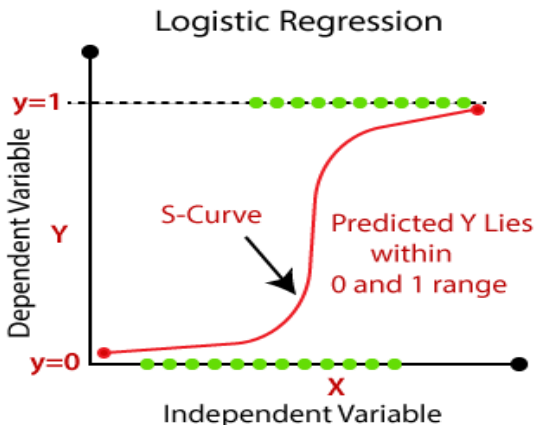
The estimated linear regression equation is: $\hat{y} = b_0 + b_1 * x_1 + b_2 * x_2$

In our example, it is $\hat{y} = -6.867 + 3.148x_1 - 1.656x_2$

Logistic Regression

- Logistic Regression is a “Supervised machine learning” algorithm that can be used to model the probability of a certain class or event. It is used when the data is linearly separable and the outcome is binary in nature.
- That means Logistic regression is usually used for Binary classification problems.
- **Logistic Regression =** $$\frac{e^{(\beta_0+\beta_1 x)}}{1+e^{(\beta_0+\beta_1 x)}}$$
- **Binary Classification** refers to predicting the output variable that is discrete in **two** classes. A few examples of Binary classification are Yes/No, Pass/Fail, Win/Lose, Cancerous/Non-cancerous, etc.





Logistic Regression

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

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

Difference between Linear Regression and Logistic Regression

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

Support Vector Machine (SVM)

Introduction

- Support Vector Machine (SVM) is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems.

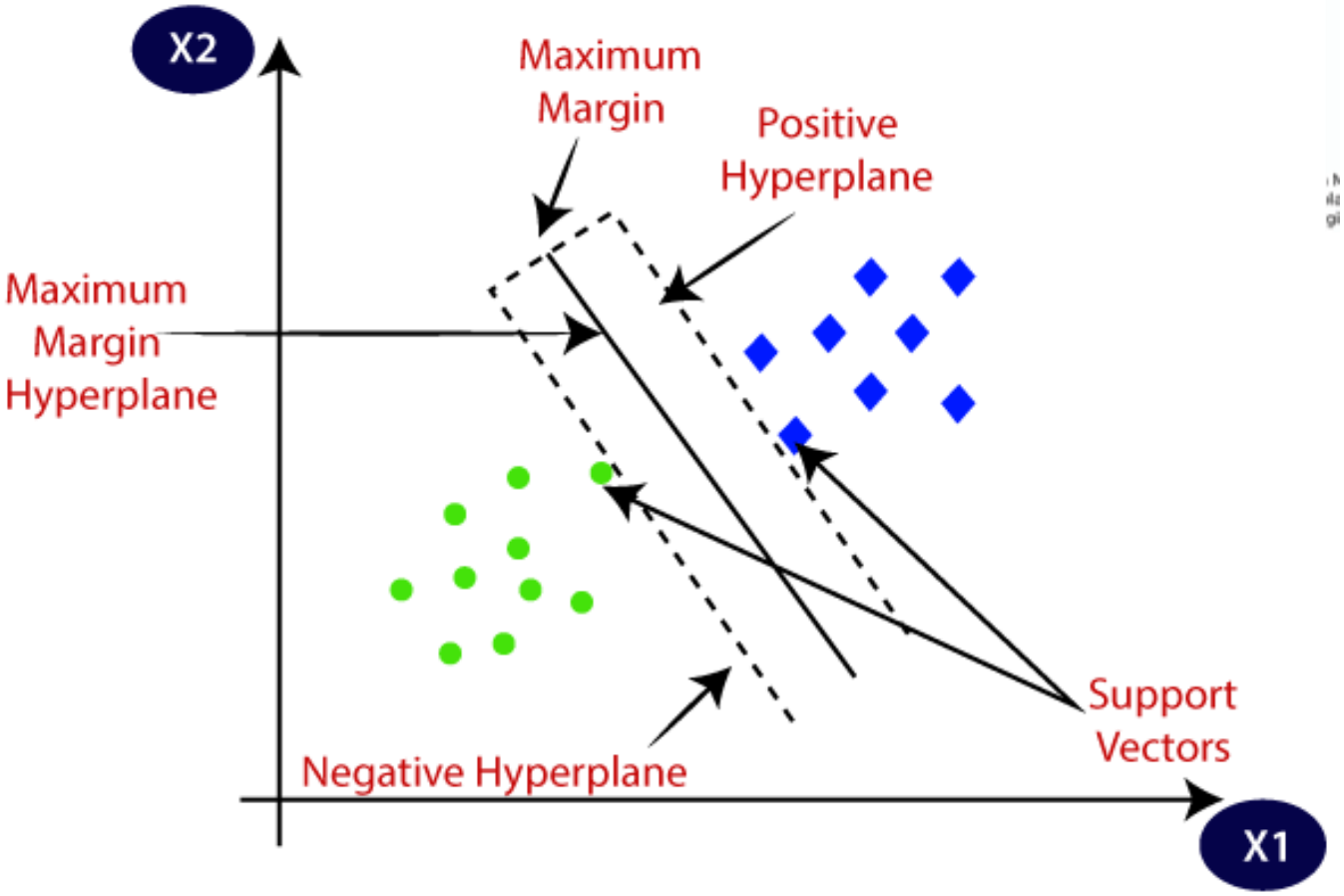
- However, primarily, it is used for Classification problems in Machine Learning.

Introduction

- SVM chooses the extreme points/vectors that help in creating the hyper-plane.
- These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

Basic Goal

- 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.


X2
Maximum
Margin
Positive
Hyperplane
Maximum
Margin
Hyperplane
Support
Vectors
Negative Hyperplane
X1

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.

Hyper-plane

- 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 hyper-plane of SVM.
- The dimensions of the hyper-plane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyper-plane will be a straight line. And if there are 3 features, then hyper-plane will be a 2-dimension plane. We always create a hyper-plane that has a maximum margin, which means the maximum distance between the data points.

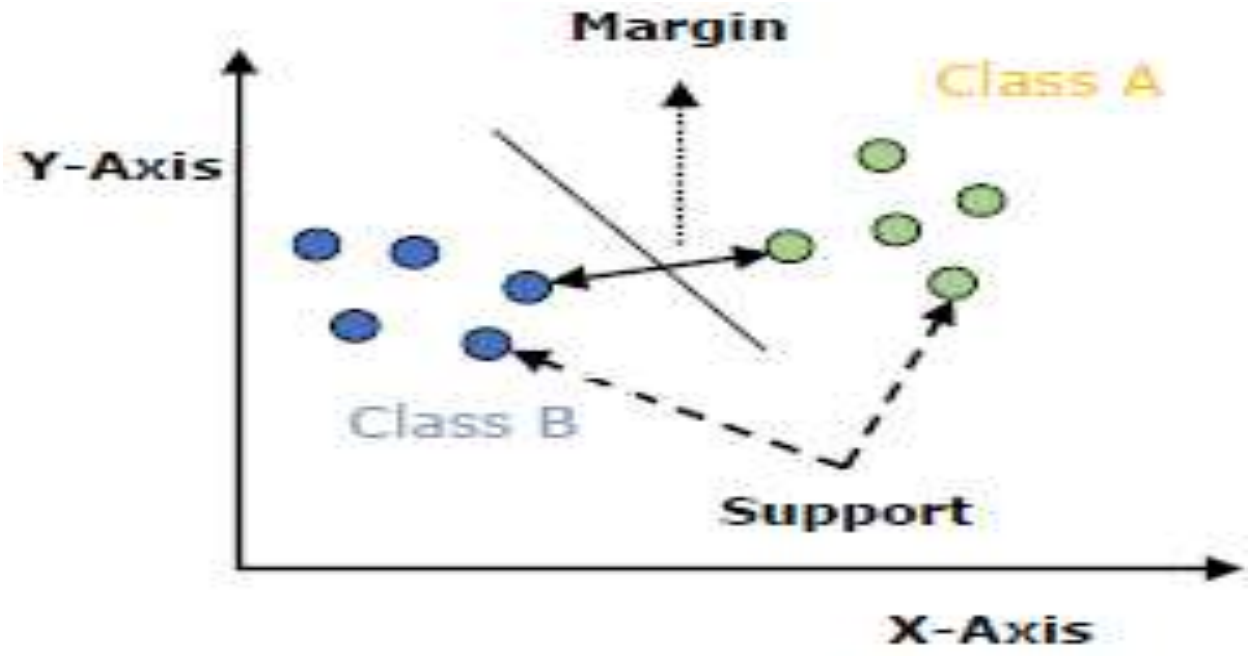
Support Vectors

- The data points or vectors that are the closest to the hyper-plane and which affect the position of the hyper-plane are termed as Support Vector.

- Since these vectors support the hyper-plane, hence called a Support vector.

Working of SVM

- An SVM model is basically a representation of different classes in a hyperplane in multidimensional space.
- The hyperplane will be generated in an iterative manner by SVM so that the error can be minimized.
- The goal is to divide the datasets into classes to find a maximum marginal hyperplane (MMH).


Margin
Class A
Y-Axis
Class B
Support
X-Axis

Concept of SVM

The followings are important concepts in SVM −

- Support Vectors
- Hyper-plane
- Margin
- **Support Vectors** − Data points that are closest to the hyper-plane is called support vectors. Separating line will be defined with the help of these data points.
- **Hyperplane** − As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.
- **Margin** − It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is considered as a bad margin.
- The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyper-plane (MMH) and it can be done in the following two steps −
  - First, SVM will generate hyper-planes iteratively that segregates the classes in best way.
  - Then, it will choose the hyper-plane that separates the classes correctly.

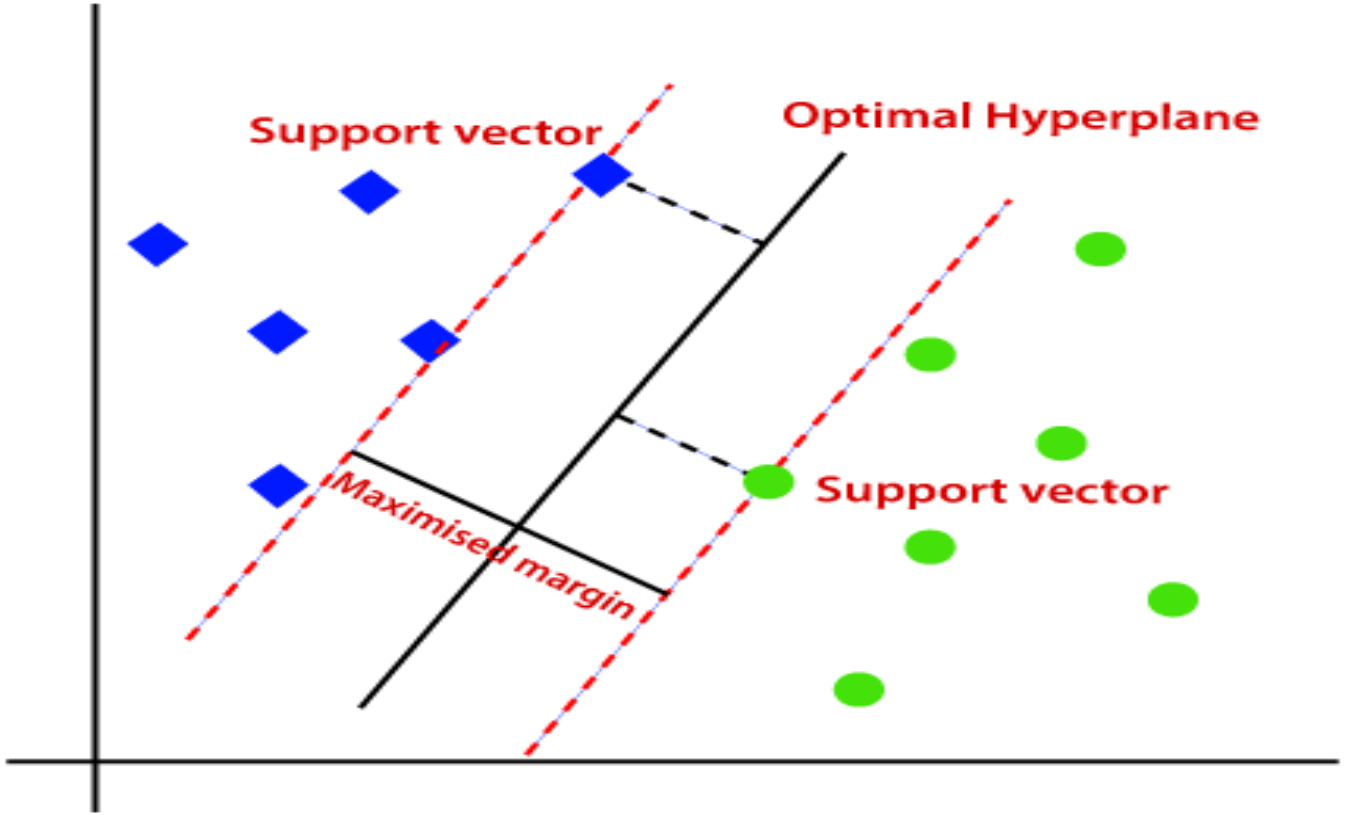
Concept of SVM

The followings are important concepts in SVM −

- Support Vectors
- Hyper-plane
- Margin
- **Support Vectors** − Data points that are closest to the hyper-plane is called support vectors. Separating line will be defined with the help of these data points.
- **Hyperplane** − As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.
- **Margin** − It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is considered as a bad margin.
- The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyper-plane (MMH) and it can be done in the following two steps −
  - First, SVM will generate hyper-planes iteratively that segregates the classes in best way.
  - Then, it will choose the hyper-plane that separates the classes correctly.

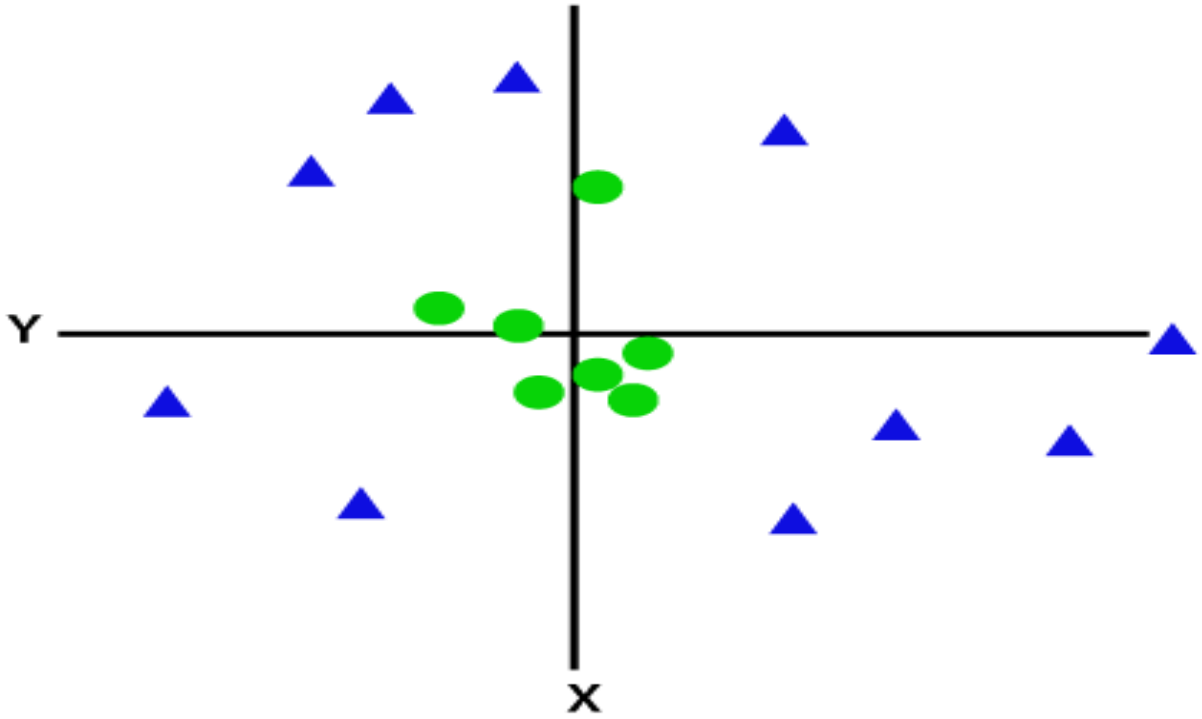
Concept of SVM

- 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**.


Support vector
Optimal Hyperplane
Maximised margin
Support vector

Working 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:

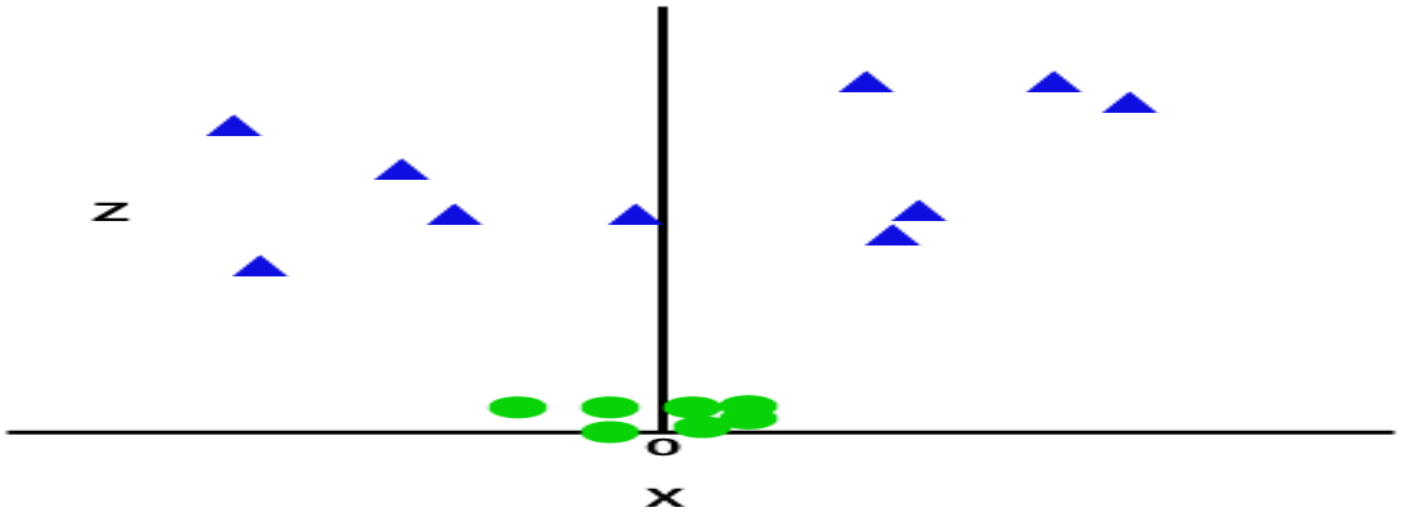


Concept of SVM

- 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:

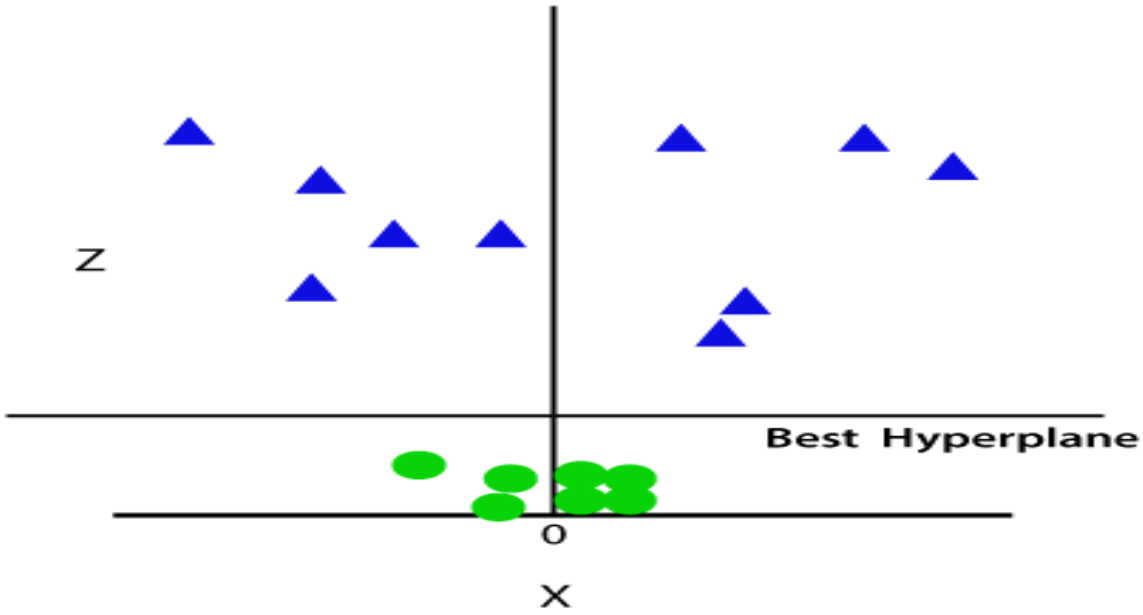
$$z=x^2+y^2$$

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



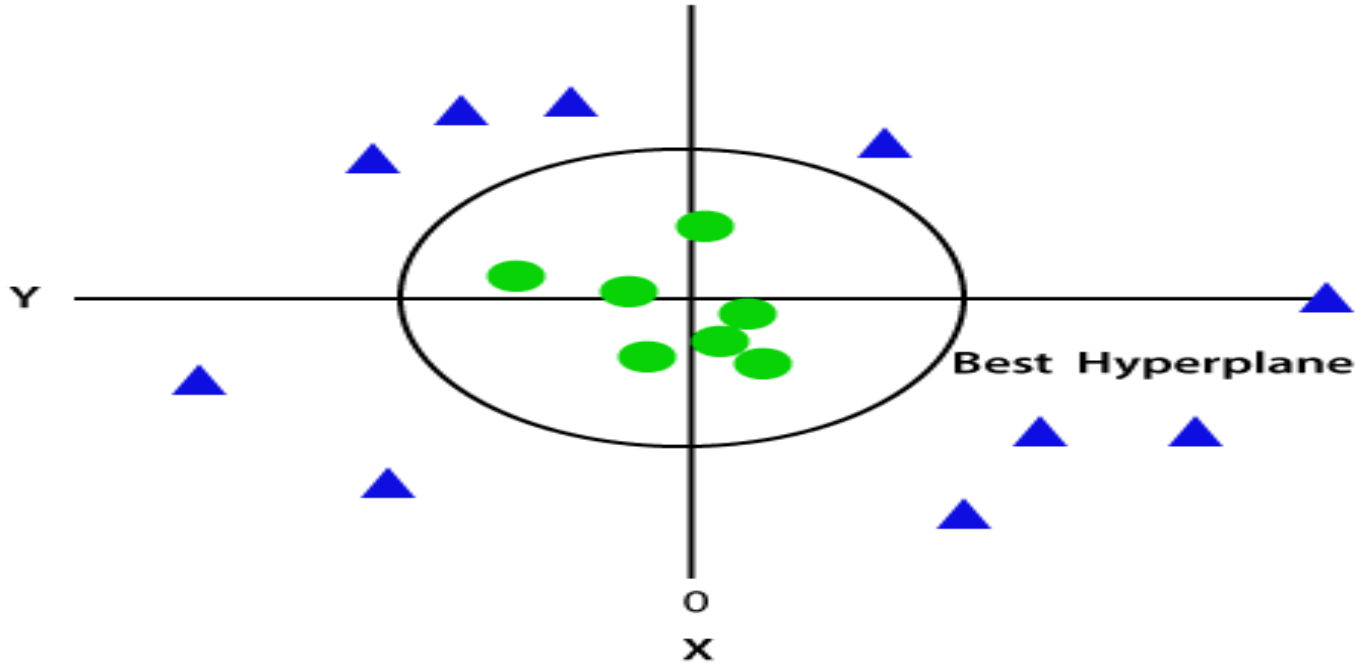
Concept of SVM

- So now, SVM will divide the datasets into classes in the following way. Consider the below image:
- Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis.
- If we convert it in 2d space with z=1, then it will become as:



Concept of 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.



SVM Pros/Cons

Pros of SVM

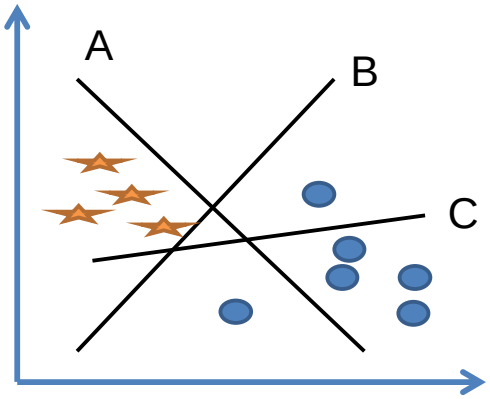
- SVM classifiers offers great accuracy and work well with high dimensional space. SVM classifiers basically use a subset of training points hence in result uses very less memory.

Cons of SVM classifiers

- They have high training time hence in practice not suitable for large datasets. Another disadvantage is that SVM classifiers do not work well with overlapping classes.

How Does Identify Right Hyperplane

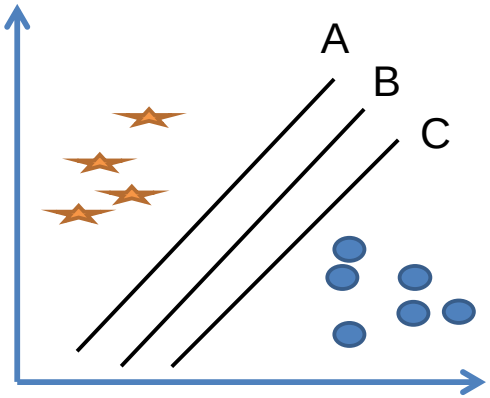
- **Scenario-1** Here we have 3 hyperplanes (A, B, and C). Now identify the right hyperplane for classify the stars and circles.



- You need to remember the thumb rule to identify the right hyperplane: select the hyperplane that segregates the two classes better.
- In this scenario hyperplane B perform their job excellently.

How Does Identify Right Hyperplane

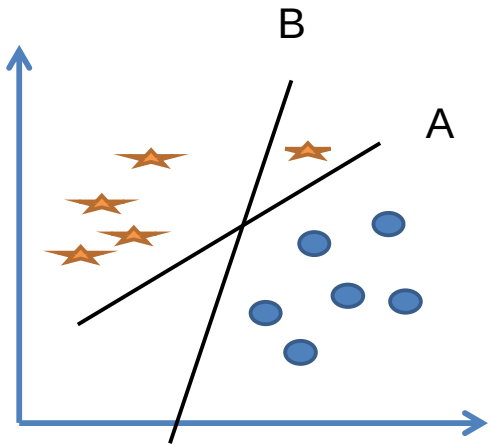
- **Scenario-2** Here we have 3 hyperplanes (A, B, and C)and all are segregating the classes well. Now how can we identify the right Hyperplane.



- Here, maximizing the distance between nearest data points and Hyperplane will help us to decide the right Hyperplane.
- In above picture margins of Hyperplane C is more than the others. So C has selected as the best Hyperplane.

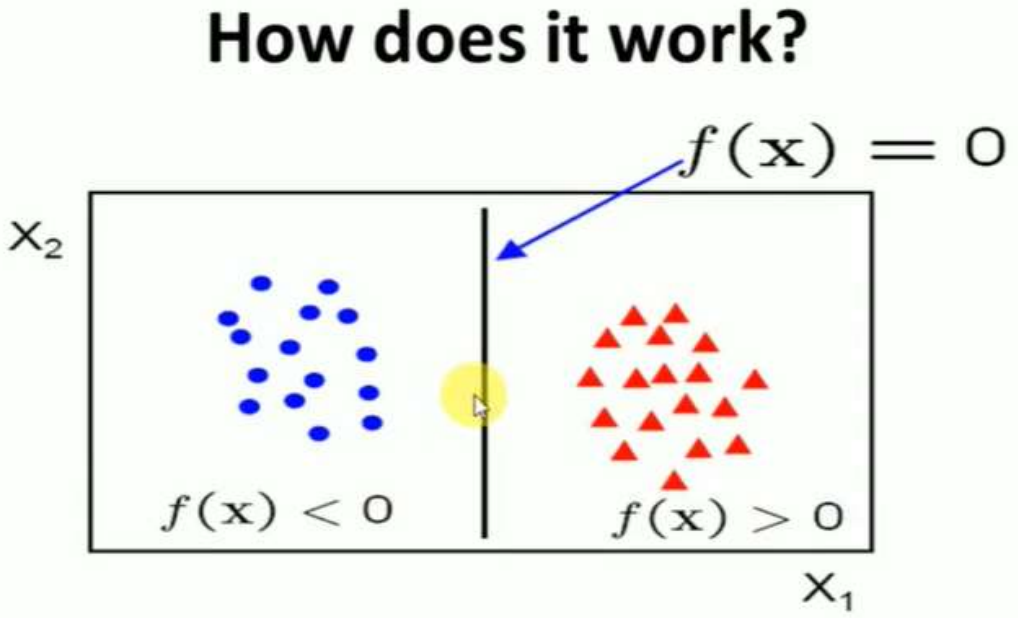
How Does Identify Right Hyperplane

- **Scenario-3** Here we have 2 hyperplanes (A, and B). Use the rules as discussed in previous section to identify the right Hyperplane



- SVM selects the Hyperplane which classifies the classes accurately prior to maximizing the margins.
- Hyperplane B has a classification error and A has classified all accurately. Therefore right Hyperplane is A.

How Does Identify Right Hyperplane



- f(x) = W.X + b
- W is the normal to the line, X is input vector and b the bias
- W is known as the weight vector

Advantages of SVM

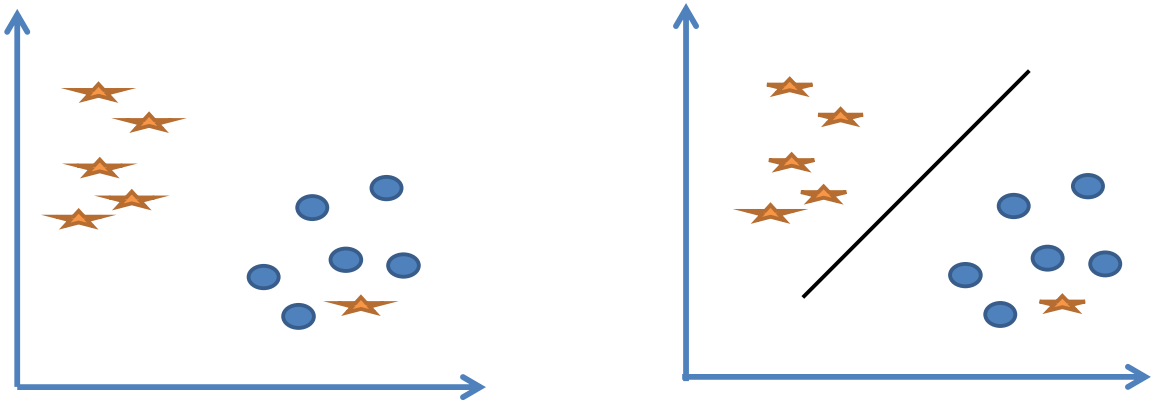
- The main strength of SVM is that they work well even when the number of SVM features is much larger than the number of instances.
- It can work on datasets with huge feature space, such is the case in spam filtering, where a large number of words are the potential signifiers of a message being spam.
- Even when the optimal decision boundary is a nonlinear curve, the SVM transforms the variables to create new dimensions such that the representation of the classifier is a linear function of those transformed dimensions of the data.
- SVMs are conceptually easy to understand. They create an easy-to-understand linear classifier.
- SVMs are now available with almost all data analytics toolsets.

Disadvantages of SVM

- The SVM technique has two major constraints
  - It works well only with real numbers, i.e., all the data points in all the dimensions must be defined by numeric values only,
  - It works only with binary classification problems. One can make a series of cascaded SVMs to get around this constraint.
- Training the SVMs is an inefficient and time consuming process, when the data is large.
- It does not work well when there is much noise in the data, and thus has to compute soft margins.
- The SVMs will also not provide a probability estimate of classification, i.e., the confidence level for classifying an instance.

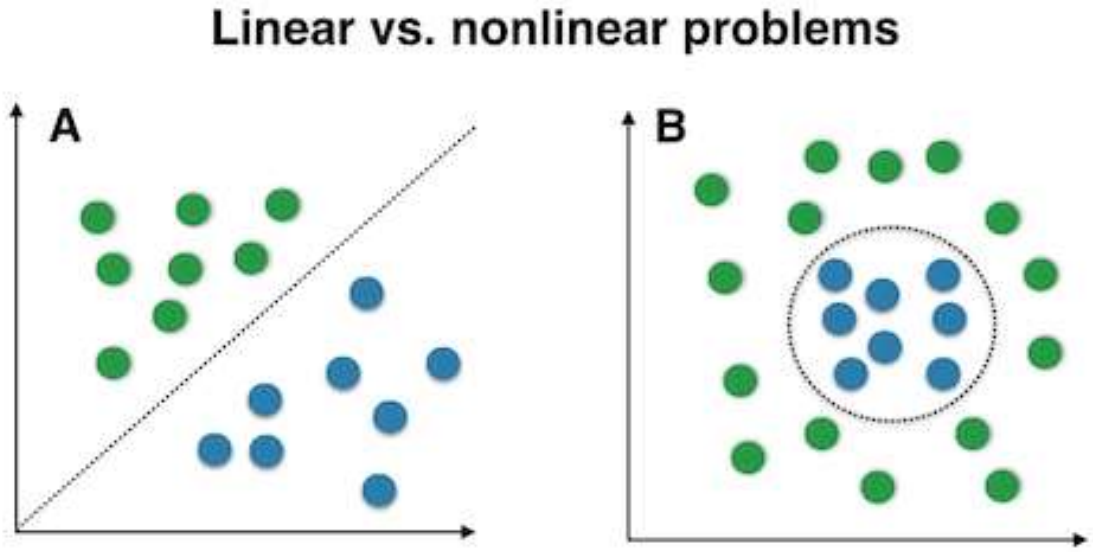
How Does Identify Right Hyperplane

- **Scenario-4** Below we are not able to segregate the two classes using straight line, as one of the star lies on the territory of circle.



- One star at other end is like an outlier for star class. The svm algorithm has the feature of ignoring the outlier and find the Hyperplane with maximum margins. Hence we can say that the svm classification is robust to outliers.

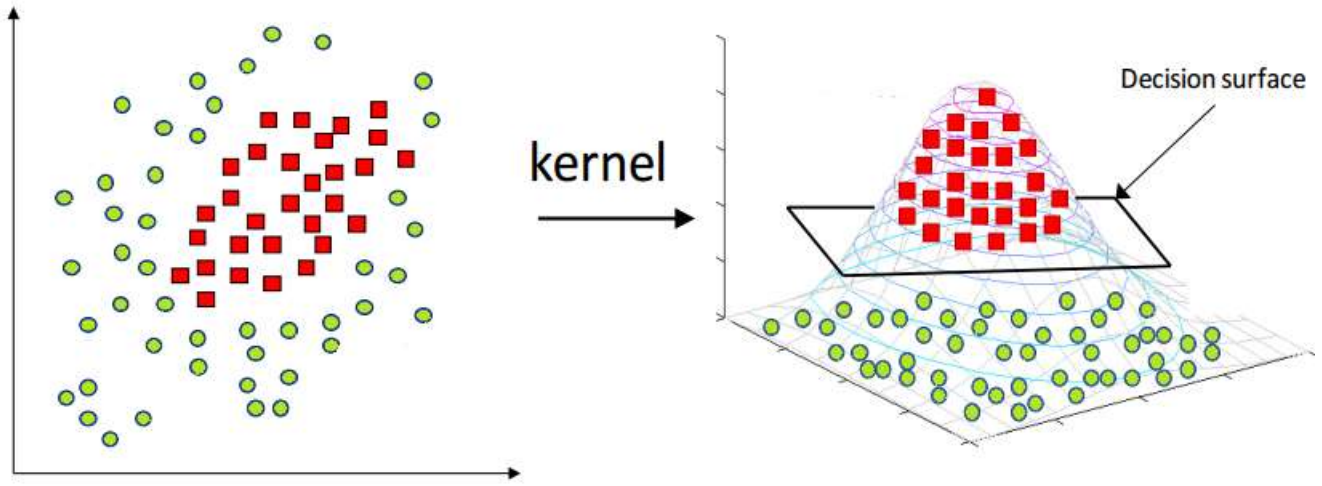
Non-Linear SVM



It is a form of data structure where the data elements don't stay arranged linearly or sequentially. Since the data structure is non-linear, it does not involve a single level.

Kernel Function

In machine learning, a kernel refers to a method that allows us to apply linear classifiers to non-linear problems by mapping non-linear data into a higher-dimensional space without the need to visit or understand that higher-dimensional space.



Kernel Trick

Kernel Trick

- Kernel trick means replacing the dot product in mapping functions with a kernel function.

$$k(x, y) = \emptyset(x) \cdot \emptyset(y)$$

- Similar to mapping functions, kernels help in mapping data from input space to higher-dimensional feature space with least computations.
- Performing the kernel operation is much easier compared to mapping functions.
- This is illustrated in the following numerical example.

Types of Kernel Function

- Polynomial Kernel
- RBF Kernel
- Sigmoid Kernel

Polynomial Kernel

- The polynomial kernel is a kernel function commonly used with support vector machines and other kernelized models, that represents the similarity of vectors in a feature space over polynomials of the original variables
- It is popular in image processing.

$$K(x_i, x_j) = (x_i^T . x_j + 1)^d$$

- Where d is the degree of polynomial.
- T is the transpose.

Radial-Basis Kernel Function

- It is general purpose kernel: used when their is no prior knowledge about the data.

$$K(x, y) = \exp\left(\frac{-|x - y|^2}{2\sigma^2}\right)$$

Hyperbolic Tangent Kernel

- Mainly used in neural networks.

$$K(x_i, x_j) = \tanh\left(kx_i.x_j + c\right)$$

- Where k>0 and c<0

SVM Solved Numerical

Support Vector Machine - Linear Example Solved

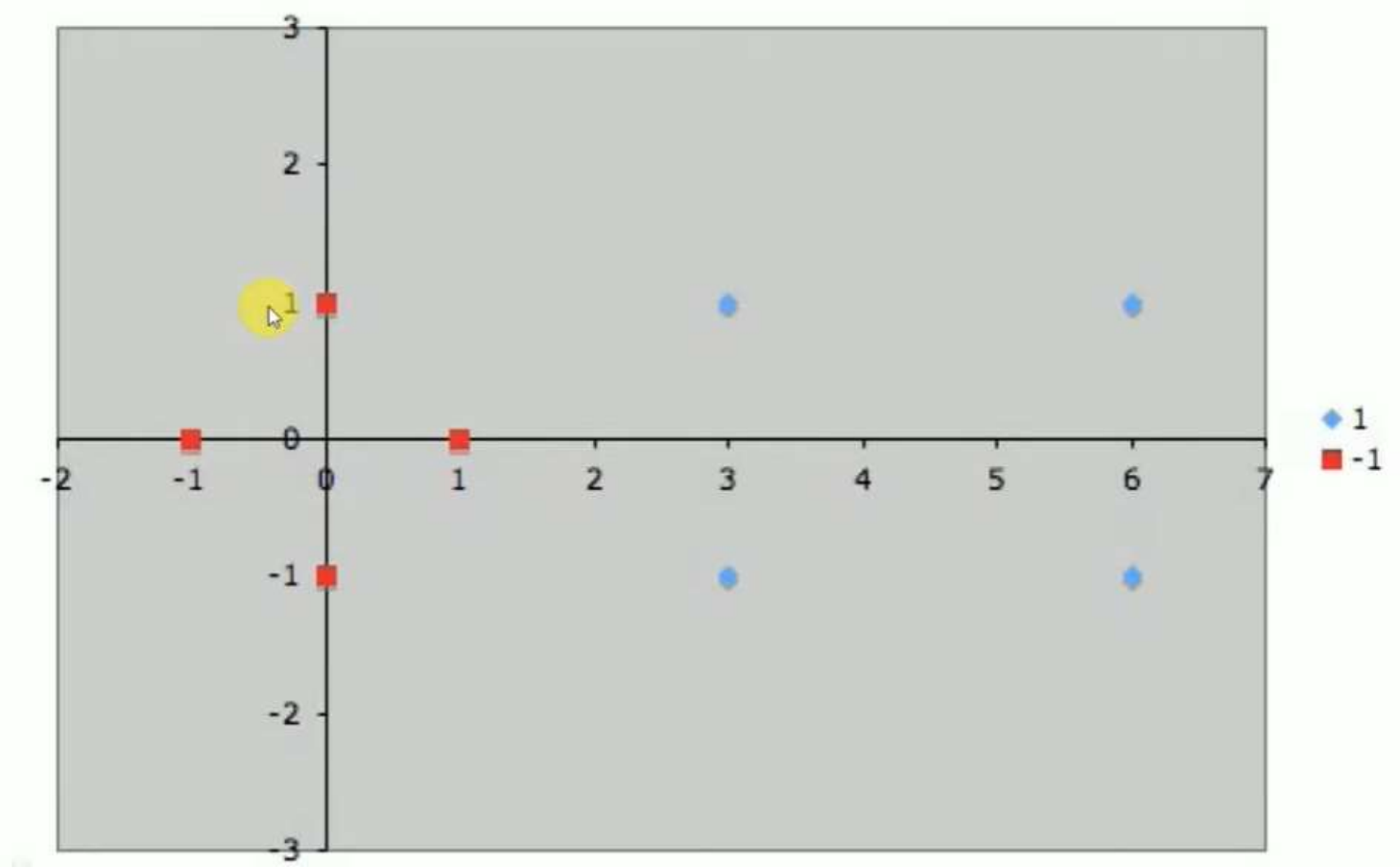
Suppose we are given the following positively labeled data points,

$$\left\{\begin{pmatrix} 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ -1 \end{pmatrix}\right\}$$

and the following negatively labeled data points,

$$\left\{\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \end{pmatrix}\right\}$$

SVM Solved Numerical



SVM Solved Numerical

- Each vector is augmented with a 1 as a bias input

- So, $s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, then $\tilde{s_1} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$

- Similarly,

- $s_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}$, then $\tilde{s_2} = \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix}$ and $s_3 = \begin{pmatrix} 3 \\ -1 \end{pmatrix}$, then $\tilde{s_3} = \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix}$

SVM Solved Numerical

$$\alpha_1 \tilde{s_1} \cdot \tilde{s_1} + \alpha_2 \tilde{s_2} \cdot \tilde{s_1} + \alpha_3 \tilde{s_3} \cdot \tilde{s_1} = -1$$
$$\alpha_1 \tilde{s_1} \cdot \tilde{s_2} + \alpha_2 \tilde{s_2} \cdot \tilde{s_2} + \alpha_3 \tilde{s_3} \cdot \tilde{s_2} = +1$$
$$\alpha_1 \tilde{s_1} \cdot \tilde{s_3} + \alpha_2 \tilde{s_2} \cdot \tilde{s_3} + \alpha_3 \tilde{s_3} \cdot \tilde{s_3} = +1$$

$$\alpha_1(1+0+1)+\alpha_2(3+0+1)+\alpha_3(3+0+1) = -1$$
$$\alpha_1(3+0+1)+\alpha_2(9+1+1)+\alpha_3(9-1+1) = 1$$
$$\alpha_1(3+0+1)+\alpha_2(9-1+1)+\alpha_3(9+1+1) = 1$$

$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix} = -1$$

$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix} = 1$$

$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}3\\-1\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}3\\-1\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}3\\-1\\1\end{pmatrix} = 1$$

$$2\alpha_1 + 4\alpha_2 + 4\alpha_3 = -1$$
$$4\alpha_1 + 11\alpha_2 + 9\alpha_3 = 1$$
$$4\alpha_1 + 9\alpha_2 + 11\alpha_3 = 1$$

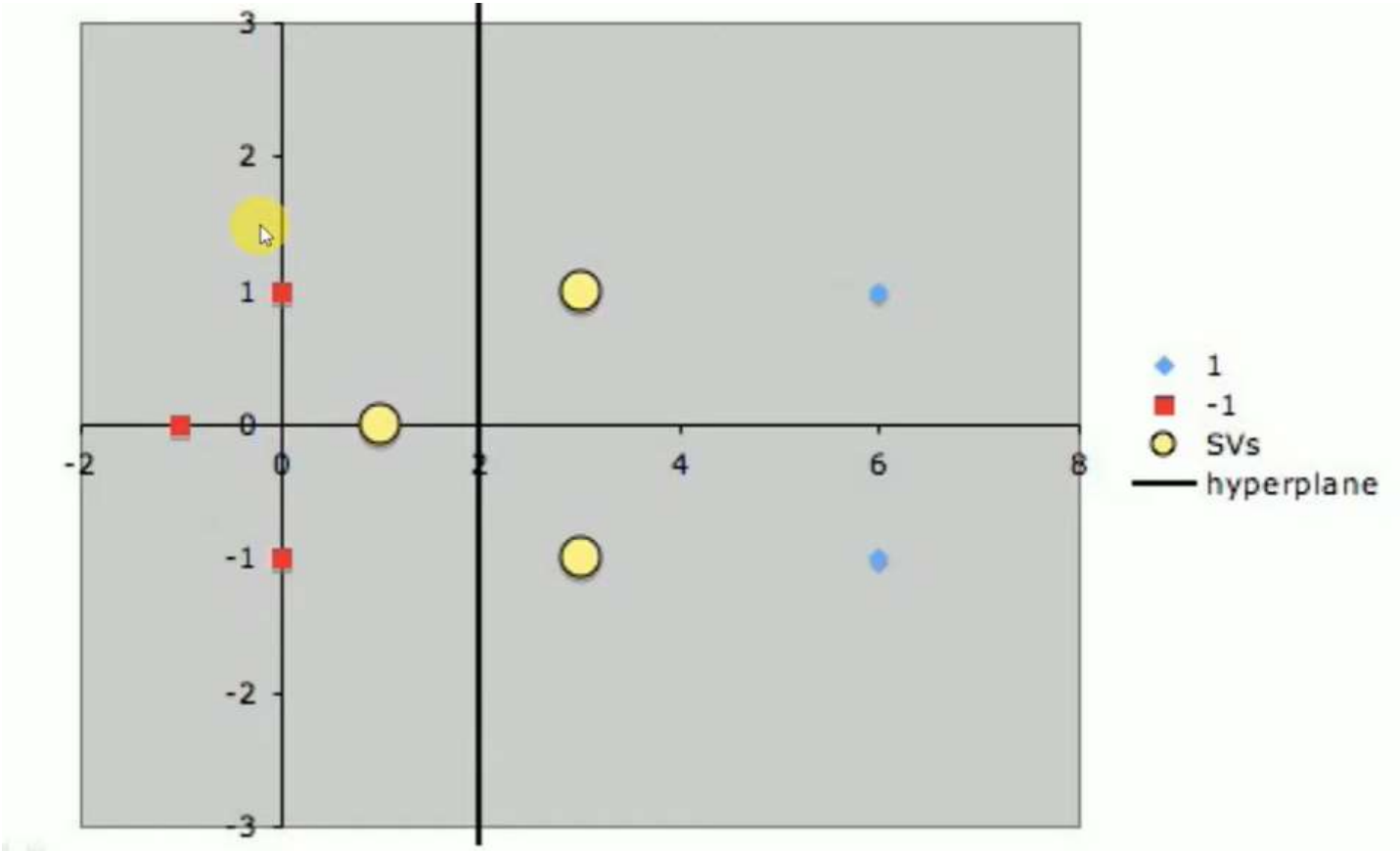
$$\alpha_1 = -3.5$$
$$\alpha_2 = 0.75$$
$$\alpha_3 = 0.75$$

SVM Solved Numerical

$$\begin{aligned}\tilde{w} &= \sum_i \alpha_i \tilde{s}_i \\ &= -3.5\begin{pmatrix}1\\0\\1\end{pmatrix} + 0.75\begin{pmatrix}3\\1\\1\end{pmatrix} + 0.75\begin{pmatrix}3\\-1\\1\end{pmatrix} \\ &= \begin{pmatrix}1\\0\\-2\end{pmatrix}\end{aligned}$$

- Finally, remembering that our vectors are augmented with a bias.
- We can equate the last entry in $\tilde{w}$ as the hyperplane offset b and write the separating
- Hyperplane equation $\boldsymbol{y = wx + b}$
- with $\boldsymbol{w} = \binom{1}{0}$ and $\boldsymbol{b = -2}$.

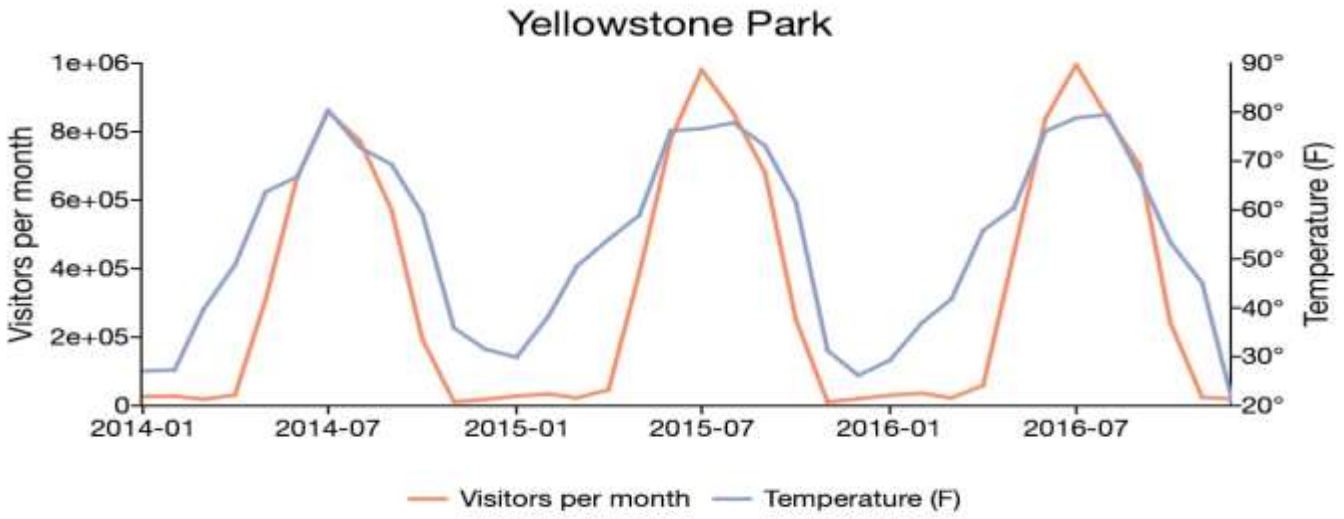
SVM Solved Numerical



Time Series Analysis

Introduction

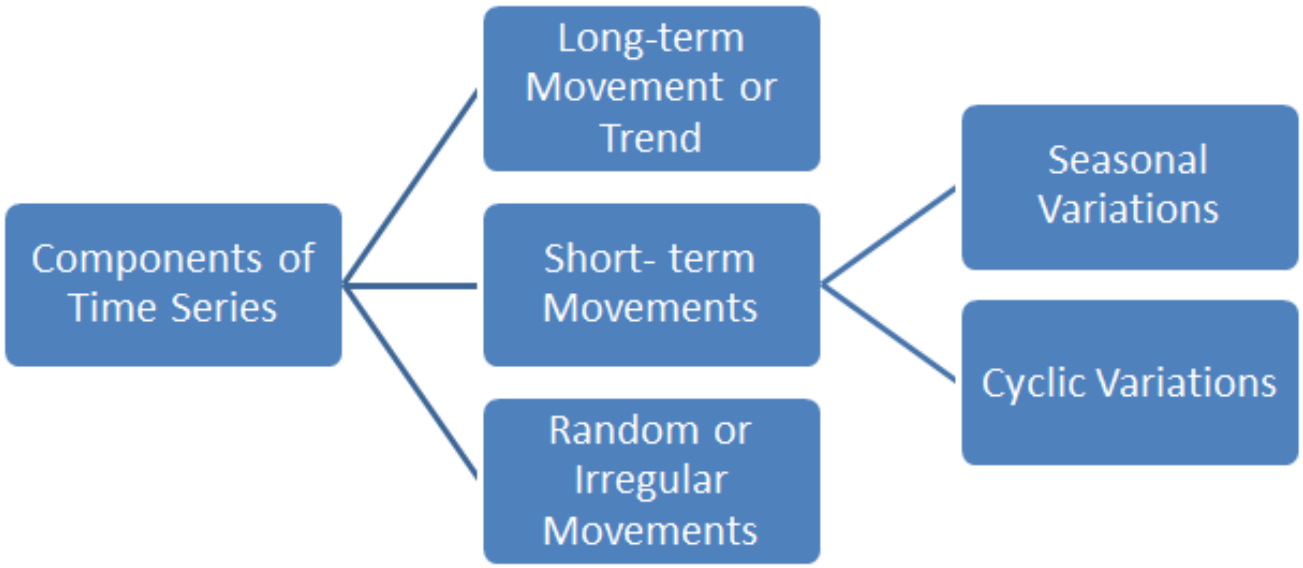
- Time series analysis is a specific way of analyzing a sequence of data points collected over an interval of time. In time series analysis, analysts record data points at consistent intervals over a set period of time rather than just recording them.
- Time series analysis helps organizations understand the underlying causes of trends or systemic patterns over time. Using data visualizations, business users can see seasonal trends and dig deeper into why these trends occur data points intermittently or randomly.



Applications of Time Series Analysis

- Weather data
- Rainfall measurements
- Temperature readings
- Heart rate monitoring (EKG)
- Brain monitoring (EEG)
- Quarterly sales
- Stock prices
- Automated stock trading
- Industry forecasts
- Interest rates

Components for Time Series Analysis



Trend

- The trend shows the general tendency of the data to increase or decrease during a long period of time. A trend is a smooth, general, long-term, average tendency. It is not always necessary that the increase or decrease is in the same direction throughout the given period of time.

- **Linear and Non-Linear Trend**
- If we plot the time series values on a graph in accordance with time t. The pattern of the data clustering shows the type of trend. If the set of data cluster more or less round a straight line, then the trend is linear otherwise it is non-linear (Curvilinear).

Periodic Fluctuations

- There are some components in a time series which tend to repeat themselves over a certain period of time. They act in a regular spasmodic manner.
- **Seasonal Variations**
- These are the rhythmic forces which operate in a regular and periodic manner over a span of less than a year. They have the same or almost the same pattern during a period of 12 months. This variation will be present in a time series if the data are recorded hourly, daily, weekly, quarterly, or monthly.
- **Cyclic Variations**
- The variations in a time series which operate themselves over a span of more than one year are the cyclic variations.

Random or Irregular Movements

- There is another factor which causes the variation in the variable under study. They are not regular variations and are purely random or irregular. These fluctuations are unforeseen, uncontrollable, unpredictable, and are erratic. These forces are earthquakes, wars, flood, famines, and any other disasters.

Mathematical Model for Time Series Analysis

- Mathematically, a time series is given as
- $y_t = f(t)$
- Here, y_t is the value of the variable under study at time t. If the population is the variable under study at the various time period t_1, t_2, t_3, … , t_n. Then the time series is

- t: $t_1, t_2, t_3, \dots, t_n$
- y_t: $y_{t1}, y_{t2}, y_{t3}, \dots, y_{tn}$

Additive Model for Time Series Analysis

- If y_t is the time series value at time t. T_t, S_t, C_t, and R_t are the trend value, seasonal, cyclic and random fluctuations at time t respectively. According to the Additive Model, a time series can be expressed as
- $y_t = T_t + S_t + C_t + R_t$.
- This model assumes that all four components of the time series act independently of each other.

Multiplicative Model for Time Series Analysis

- The multiplicative model assumes that the various components in a time series operate proportionately to each other. According to this model
- $y_t = T_t \times S_t \times C_t \times R_t$

Mixed Models

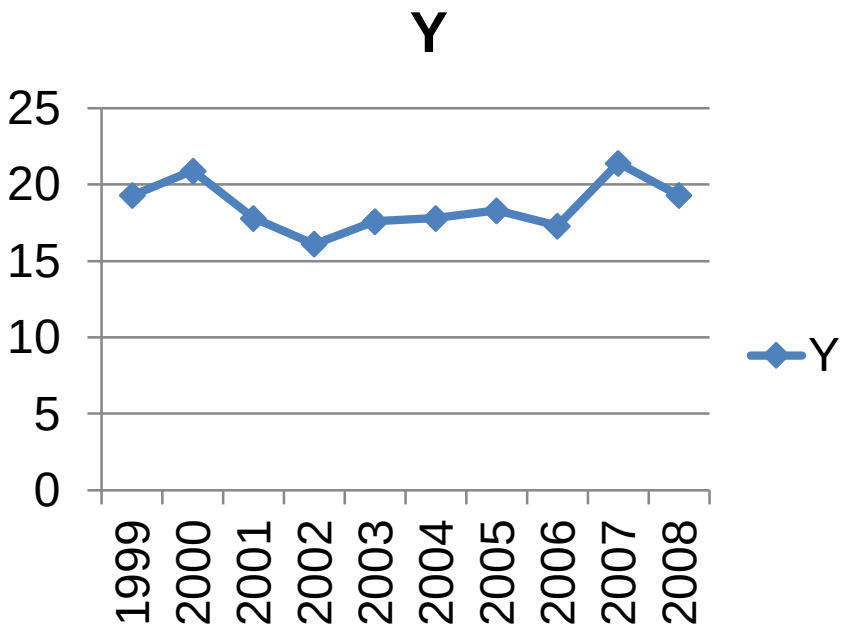
- Different assumptions lead to different combinations of additive and multiplicative models as
- $y_t = T_t + S_t + C_t R_t$.
- The time series analysis can also be done using the model
- $y_t = T_t + S_t \times C_t \times R_t$ or $y_t = T_t \times C_t + S_t \times R_t$ etc.

Time Series Analysis Methods

1. Graphical Method
2. Moving Average Method
3. Least Square Method

Graphical Method

Year	Y
1999	19.3
2000	20.9
2001	17.8
2002	16.1
2003	17.6
2004	17.8
2005	18.3
2006	17.3
2007	21.4
2008	19.3



Moving Average Method

One method of establishing the underlying trend (smoothing out peaks and troughs) in a set of data is using the **moving averages technique.** Other methods, such as regression analysis can also be used to estimate the trend.

A moving average is a series of averages, calculated from historic data. Moving averages can be calculated for any number of time periods, for example a three-month moving average, a seven-day moving average, or a four-quarter moving average.

Month	Sales ($000)	Three-month moving total ($000)	Three-month moving average ($000)	Seasonal variation ($000)
January	125			
February	145	456=(125+145+186)	(456 ÷ 3) = 152	
March	186	462=(145+186+131)	(462 ÷ 3) = 154	
April	131	468=(186+131+151)	(468 ÷ 3) = 156	
May	151	474	158	
June	192	480	160	
July	137	486	162	
August	157	492	164	
September	198	498	166	
October	143	504	168	
November	163	510	170	
December	204			

Moving Average Method

- **Calculate the trend:**
- The three-month moving average represents the **trend**. From our example we can see a clear trend in that each moving average is $2,000 higher than the preceding month moving average. This suggests that the sales revenue for the company is, on average, growing at a rate of $2,000 per month.

Moving Average Method

- **Calculate the seasonal variation:**
- Once a trend has been established, any **seasonal variation** can be calculated. The seasonal variation can be assumed to be the difference between the actual sales and the trend (three-month moving average) value. Seasonal variations can be calculated using the additive or multiplicative models.

Month	Sales ($000)	Three-month moving total ($000)	Three-month moving average ($000)	Seasonal variation ($000)
January	125			
February	145	456	152	(145 – 152) = -7
March	186	462	154	(186 – 154) = 32
April	131	468	156	(131 – 156) = -25
May	151	474	158	(151 – 158) = -7
June	192	480	160	(192 – 160) = 32
July	137	486	162	(137 – 162) = -25
August	157	492	164	(157 – 164) = -7
September	198	498	166	(198 – 166) = 32
October	143	504	168	(143 – 168) = -25
November	163	510	170	(163 – 170) = -7
December	204			

Moving Average Method

- From the data we can see a clear three-month cycle in the seasonal variation. Every first month has a variation of -7, suggesting that this month is usually $7,000 below the average. Every second month has a variation of 32 suggesting that this month is usually $32,000 above the average. In month 3, the variation suggests that every third month, the actual will be $25,000 below the average.

Least Square Method

- The least-squares principle says that “the sum of squares of the deviations of the observed values from the corresponding expected values should be least”. Among all the trend lines, the trend line is called a least-squares fit for which the sum of the squares of the deviations of the observed values form their corresponding expected values is the least.

Year	Sales	x=x-a	x^2	xy	Y=a+bx
2015	30	-2	4	-60	45
2016	50	-1	1	-50	50
2017	75	0	0	0	55
2018	80	1	1	80	60
2019	40	2	4	80	65
N=5	$\sum y=275$		$\sum x^2=10$	$\sum xy=50$	

$$y = a + bx$$

$$a = \frac{\Sigma y}{N} = \frac{275}{5} = 55$$

$$b = \frac{\Sigma xy}{x^2} = \frac{50}{10} = 5$$

Rule Induction

1. Rule induction is a data mining process of deducing if-then rules from a dataset.

2. These symbolic decision rules explain an inherent relationship between

the attributes and class labels in the dataset.

3. Many real-life experiences are based on intuitive rule induction.

4. Rule induction provides a powerful classification approach that can be easily understood by the general users.

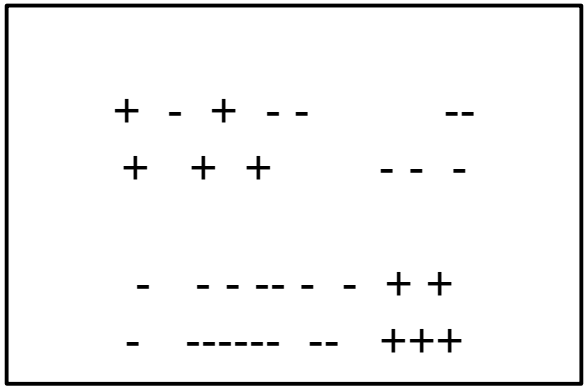
5. It is used in predictive analytics by classification of unknown data.

6. Rule induction is also used to describe the patterns in the data.

7. The easiest way to extract rules from a data set is from a decision tree that is developed on the same data set.

Procedure of Extracting Rules

- **Step 1 : Class selection :**
- a. The algorithm starts with selection of class labels one by one.
- b. The rule set is class-ordered where all the rules for a class are developed before moving on to next class.
- c. The first class is usually the least-frequent class label.
- d. From Fig. 1, the least frequent class is “+” and the algorithm focuses on generating all the rules for “+” class.



Procedure of Extracting Rules

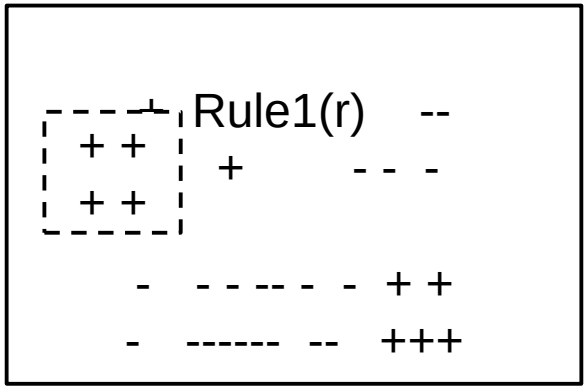
- **Step 2: Rule development :**

a. The objective in this step is to cover all "+" data points using classification rules with none or as few "–" as possible.

b. For example, in Fig. 2.10.2 , rule *r*1 identifies the area of four "+" in the top left corner.

C. Since this rule is based on simple logic operators in conjuncts, the boundary is rectilinear.

d. Once rule r1 is formed, the entire data points covered by r1 are eliminated and the next best rule is found from data sets.



Procedure of Extracting Rules

- **Step 3 : Learn-One-Rule :**

a. Each rule *r*1 is grown by the learn-one-rule approach.

b. Each rule starts with an empty rule set and conjuncts are added one by one to increase the rule accuracy.

c. Rule accuracy is the ratio of amount of "+" covered by the rule to all records covered by the rule :

$$\textit{Rule}\ \text{Accuracy}\ A(r_i) = \frac{\textit{Correct records by rule}}{\text{All records covered by the rule}}$$

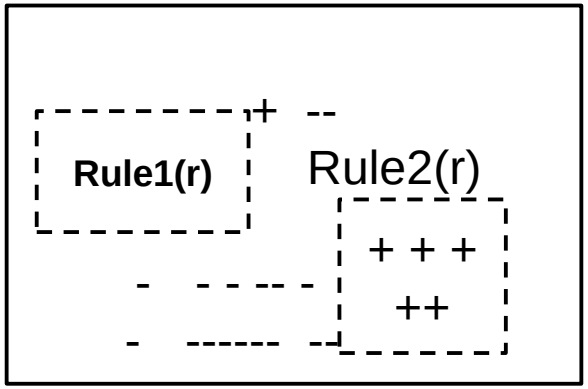
d. Learn-one-rule starts with an empty rule set: if {} then class = "+".

e. The accuracy of this rule is the same as the proportion of + data points in the data set. Then the algorithm greedily adds conjuncts until the accuracy reaches 100 %.

f. If the addition of a conjunct decreases the accuracy, then the algorithm looks for other conjuncts or stops and starts the iteration of the next rule.

Procedure of Extracting Rules

- **Step 4 : Next rule :**
- a. After a rule is developed, then all the data points covered by the
- rule are eliminated from the data set.
- b. The above steps are repeated for the next rule to cover the rest of the "+" data points.
- c. In Fig. 2.10.3, rule *r*2 is developed after the data points covered by *r*1 are eliminated.



Procedure of Extracting Rules

- **Step 5 : Development of rule set :**

a. After the rule set is developed to identify all “+” data points, the rule model is evaluated with a data set used for pruning to reduce generalization errors.

b. The metric used to evaluate the need for pruning is $(p – n)/(p + n)$, where p is the number of positive records covered by the rule and n is the number of negative records covered by the rule.

c. All rules to identify “+” data points are aggregated to form a rule

group.