SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT – I – DEEP LEARNING – SCSA3015

UNIT I INTRODUCTION TO DEEP LEARNING

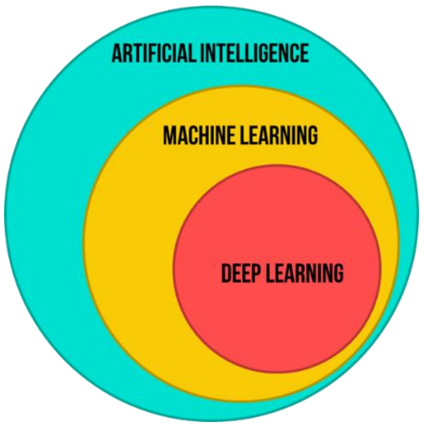
Introduction to machine learning - Linear models (SVMs and Perceptron's, logistic regression)- Introduction to Neural Nets: What are a shallow network computes- Training a network:loss functions, back propagation and stochastic gradient descent- Neural networks as universal function approximates.

1.1 <u>Introduction to Machine Learning</u>

Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.
Example: Image recognition, Speech recognition, Medical diagnosis, Statistical arbitrage, Predictive analytics, etc.

1.1.1 <u>Artificial Intelligence, Machine Learning and Deep Learning</u>

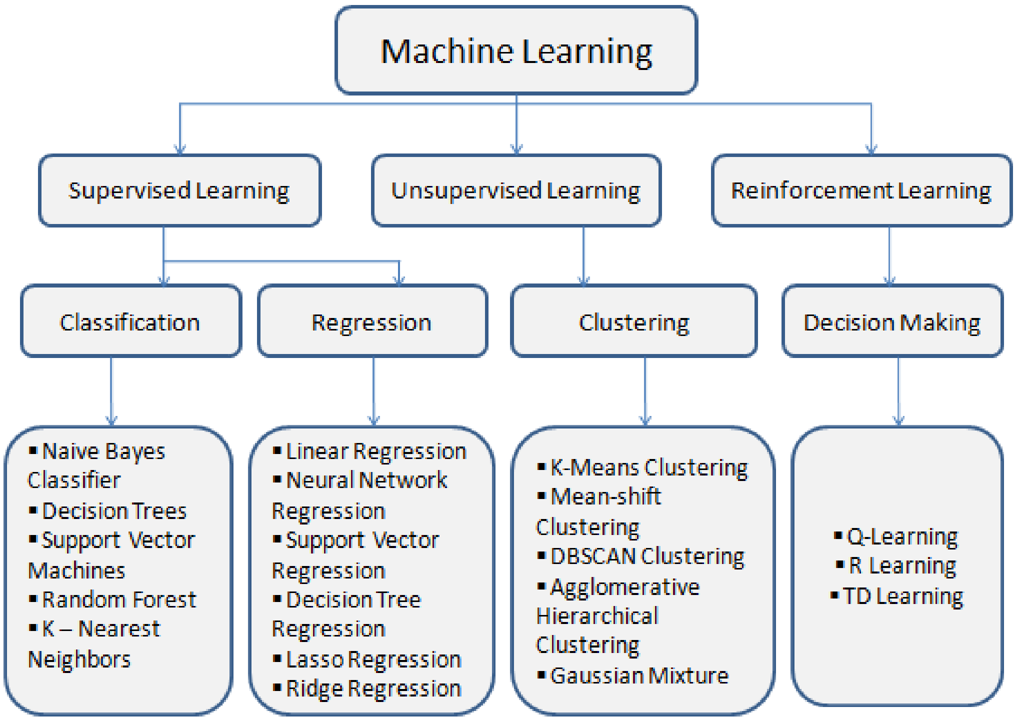
- **Artificial Intelligence** is defined as a program that exhibits cognitive ability similar to that of a human being. It makes computers think like humans and solve problems the way we do is one of the main tenets of artificial intelligence.
- Any computer program that shows characteristics, such as self-improvement, learning through inference, or even basic human tasks, such as image recognition and language processing, is considered to be a form of AI.
- The field of artificial intelligence includes within it the sub-fields of machine learning and deep learning.
- **Deep Learning** is a more specialized version of machine learning that utilizes more complex methods for difficult problems.



1.1.2. Types of Machine Learning Algorithms:

These are three types of machine learning:

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning



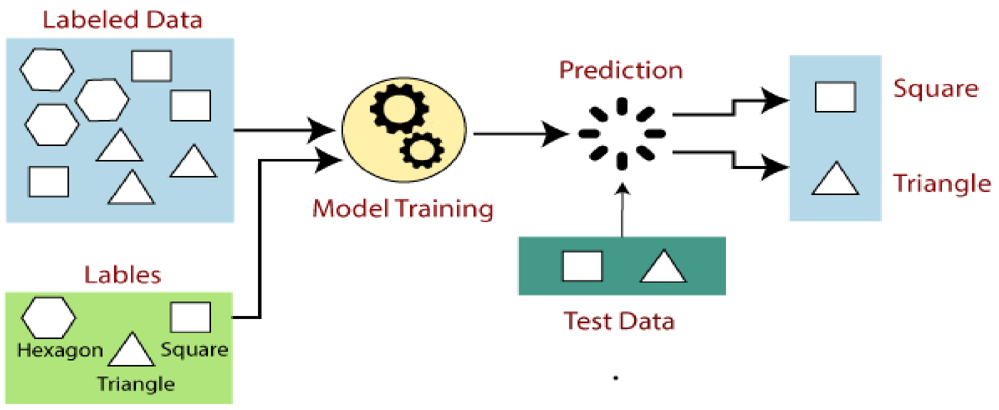
1) Supervised Learning:

- Supervised learning is one of the most basic types of machine learning.
- In this type, the machine learning algorithm is trained on **labelled data**.
- In supervised learning, the ML algorithm is given a small training dataset to work with.
- This training dataset is a smaller part of the bigger dataset and serves to give the algorithm a basic idea of the problem, solution, and data points to be dealt with.
- The algorithm then finds relationships between the parameters given, essentially establishing a cause-and-effect relationship between the variables in the dataset.
- At the end of the training, the algorithm has an idea of how the data works and the relationship between the input and the output.

- This solution is then deployed for use with the final dataset, which it learns from in the same way as the training dataset.
- Example: Risk Assessment, Image classification, Fraud Detection, spam filtering, etc.

<u>How Supervised Learning Works?</u>

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



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

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

Now, after training, we test our model using the test set, and the task of the model is to identify the shape. The machine is already trained on all types of shapes, and when it finds a new shape, it classifies the shape on the bases of a number of sides, and predicts the output.

<u>Advantages of Supervised learning:</u>

- With the help of supervised learning, the model can predict the output on the basis of prior experiences.
- In supervised learning, we can have an exact idea about the classes of objects.

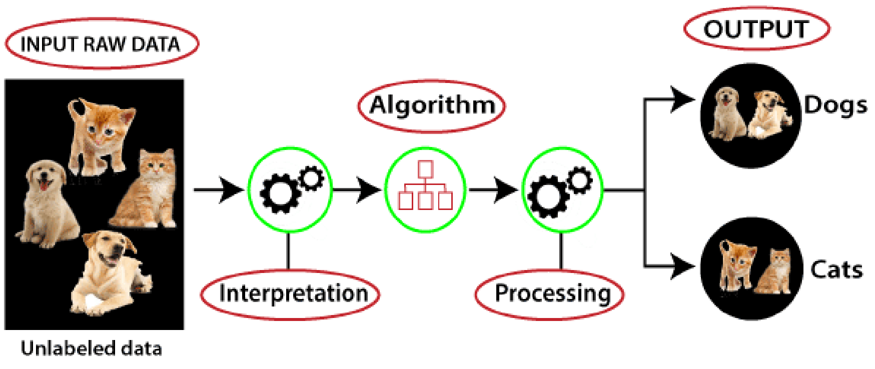
Disadvantages of supervised learning:

- Supervised learning models are not suitable for handling the complex tasks.
- Supervised learning cannot predict the correct output if the test data is different from the training dataset.
- Training required lots of computation times.

2) Unsupervised Learning:

- Unsupervised machine learning holds the advantage of being able to work with **unlabelled data**.
- This means that human labour is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program.
- In supervised learning, the labels allow the algorithm to find the exact nature of the relationship between any two data points. However, unsupervised learning does not have labels to work off of, resulting in the creation of hidden structures.
- Relationships between data points are perceived by the algorithm in an abstract manner, with no input required from human beings.
- The creation of these hidden structures is what makes unsupervised learning algorithms versatile.
- Instead of a defined and set problem statement, unsupervised learning algorithms can adapt to the data by dynamically changing hidden structures.
- This offers more post-deployment development than supervised learning algorithms.
- Example : Principal Component Analysis, Clustering

How Unsupervised Learning Works?



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

Advantages of Unsupervised Learning

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

Disadvantages of Unsupervised Learning

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

3) Semi-Supervised learning

- It is a type of Machine Learning algorithm that represents the intermediate ground between Supervised and Unsupervised learning algorithms. It uses the combination of labeled and unlabeled datasets during the training period.
- The basic disadvantage of supervised learning is that it requires hand-labeling by ML specialists or data scientists, and it also requires a high cost to process. Further unsupervised learning also has a limited spectrum for its applications. To overcome these drawbacks of supervised learning and unsupervised learning algorithms, the concept of Semi-supervised learning is introduced**.**

Applications:

- Speech Analysis
- Protein sequence classification
- Text document classifier

4) Reinforcement Learning

- Reinforcement Learning directly takes inspiration from how human beings learn from data in their lives. It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favourable outputs are encouraged or 'reinforced', and non-favourable outputs are discouraged or 'punished'.
- In every iteration of the algorithm, the output result is given to the interpreter, which decides whether the outcome is favourable or not.
- In case of the program finding the correct solution, the interpreter reinforces the solution by providing a reward to the algorithm. If the outcome is not favourable, the algorithm is forced to reiterate until it finds a better result. In most cases, the reward system is directly tied to the effectiveness of the result.
- In typical reinforcement learning use-cases, such as finding the shortest route between two points on a map, the solution is not an absolute value. Instead, it takes on a score of effectiveness, expressed in a percentage value. The higher this percentage value is, the more reward is given to the algorithm.
- Thus, the program is trained to give the best possible solution for the best possible reward.

Types of Reinforcement learning

There are mainly two types of reinforcement learning, which are:

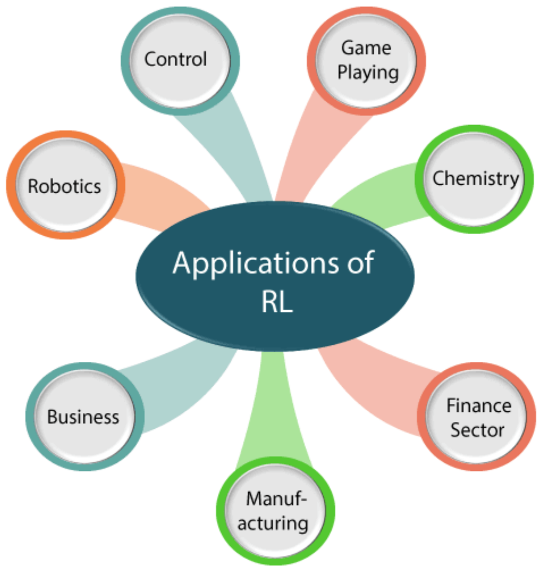
- **Positive Reinforcement**

The positive reinforcement learning means adding something to increase the tendency that expected behaviour would occur again. It impacts positively on the behaviour of the agent and increases the strength of the behaviour. This type of reinforcement can sustain the changes for a long time, but too much positive reinforcement may lead to an overload of states that can reduce the consequences.

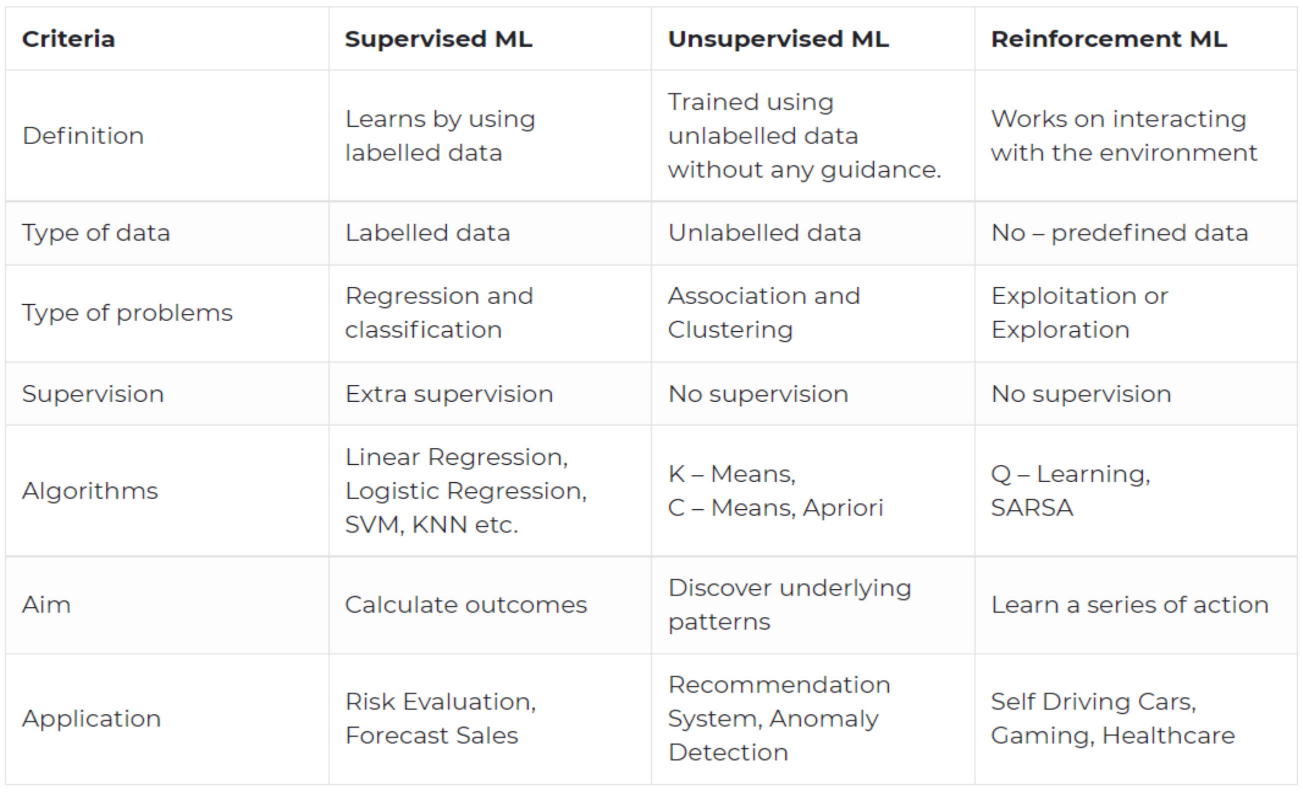
- **Negative Reinforcement:**

The negative reinforcement learning is opposite to the positive reinforcement as it increases the tendency that the specific behaviour will occur again by avoiding the negative condition. It can be more effective than the positive reinforcement depending on situation and behaviour, but it provides reinforcement only to meet minimum behaviour.

Reinforcement Learning Applications



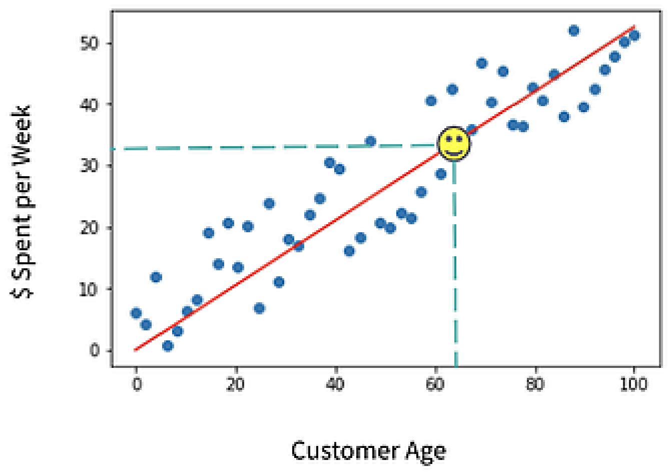
Difference Between Supervised, Unsupervised and Reinforcement Learning



Criteria	Supervised ML	Unsupervised ML	Reinforcement ML
Definition	Learns by using labelled data	Trained using unlabelled data without any guidance.	Works on interacting with the environment
Type of data	Labelled data	Unlabelled data	No – predefined data
Type of problems	Regression and classification	Association and Clustering	Exploitation or Exploration
Supervision	Extra supervision	No supervision	No supervision
Algorithms	Linear Regression, Logistic Regression, SVM, KNN etc.	K – Means, C – Means, Apriori	Q – Learning, SARSA
Aim	Calculate outcomes	Discover underlying patterns	Learn a series of action
Application	Risk Evaluation, Forecast Sales	Recommendation System, Anomaly Detection	Self Driving Cars, Gaming, Healthcare

1.2 Linear models

Linear models generate a formula to create a best-fit line to predict unknown values. Linear models are considered “old school” and often not as predictive as newer algorithm classes, but they can be trained relatively quickly and are generally more straightforward to interpret. A Simple linear model would be linear regression, and consider an example of using age to predict how much is spent in a week.



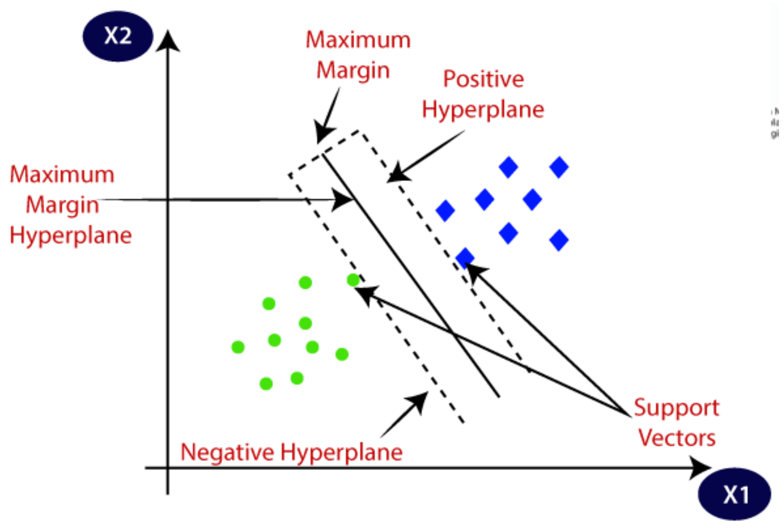
The data points we used to train our model are in blue. The red line is the line of best fit, which the model generated, and captures the direction of those points as best as possible.

1.2.1 Support Vector Machine

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

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

Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



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

<u>Hyperplane and Support Vectors in the SVM algorithm:</u>

Hyperplane:

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

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

Support Vectors:

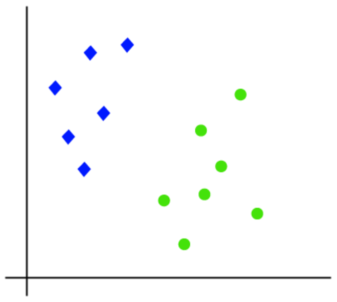
The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

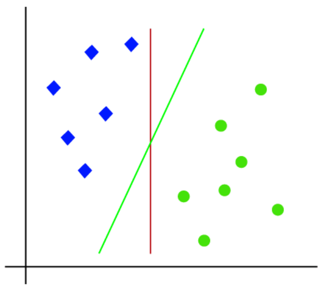
Types of SVM

- **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.

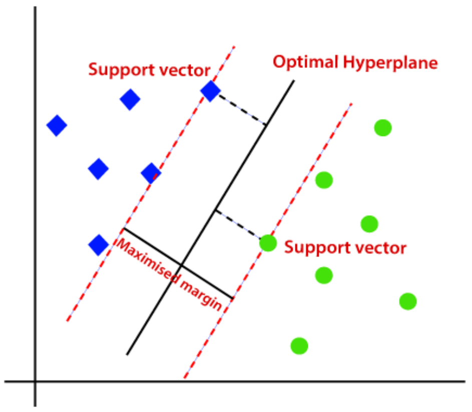
Linear SVM

The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:



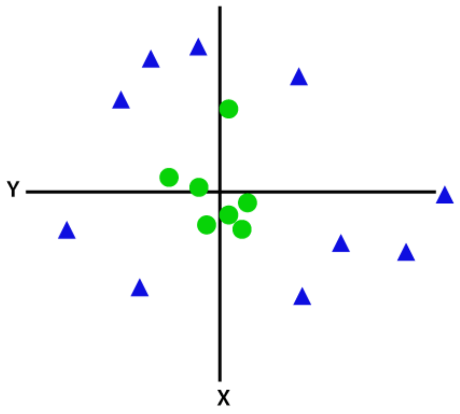


The SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a **hyperplane**. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called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



Non-Linear SVM:

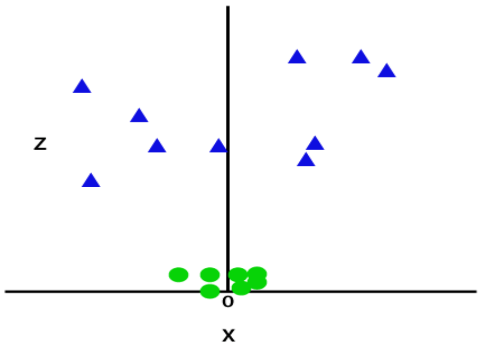
If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



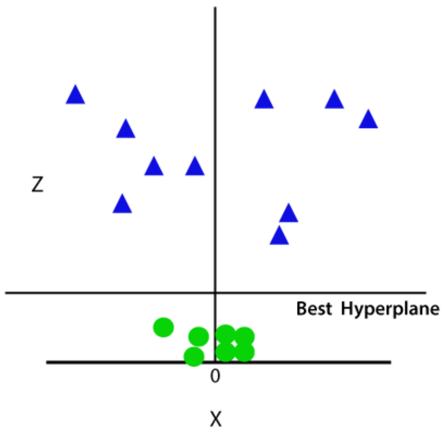
So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third-dimension z. It can be calculated as:

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

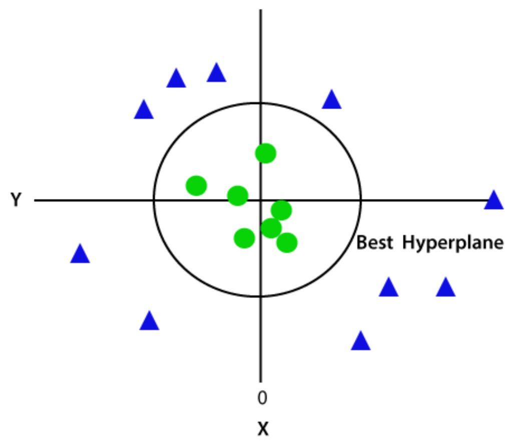
By adding the third dimension, the sample space will become as below image:



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



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



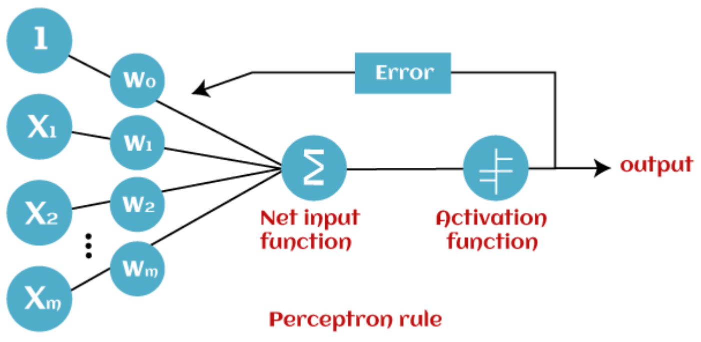
Hence we get a circumference of radius 1 in case of non-linear data.

<u>1.2.2 Perceptron</u>

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

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

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



- **<u>Input Nodes or Input Layer:</u>**

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

- **<u>Wight and Bias:</u>**

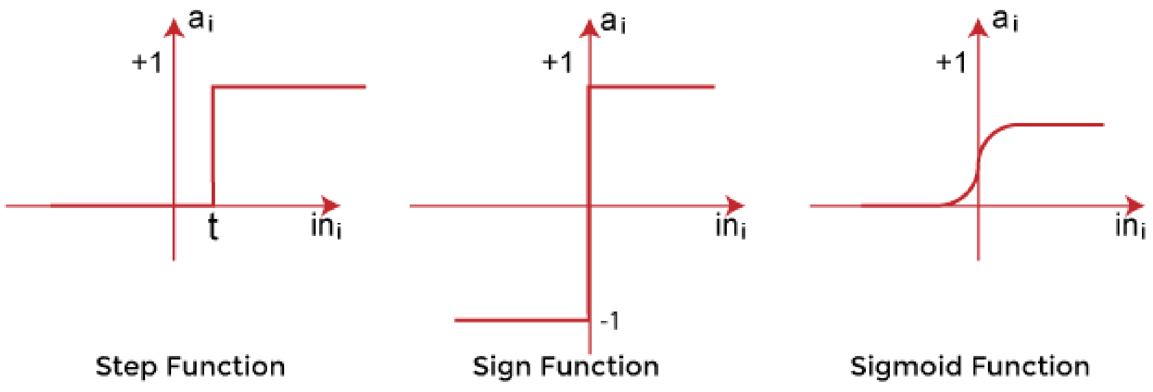
Weight parameter represents the strength of the connection between units. This is another most important parameter of Perceptron components. Weight is directly proportional to the strength of the associated input neuron in deciding the output. Further, Bias can be considered as the line of intercept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

- **<u>Activation Function:</u>**

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

<u>Types of Activation functions:</u>

- Sign function
- Step function, and
- Sigmoid function



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

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

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

Perceptron model works in two important steps as follows:

Step 1: In the first step first, multiply all input values with corresponding weight values and then add them to determine the weighted sum. Mathematically, we can calculate the weighted sum as follows:

$$\sum w_i * x_i = x_1 * w_1 + x_2 * w_2 + \ldots w_n * x_n$$

Add a special term called **bias 'b'** to this weighted sum to improve the model's performance.

$$\boldsymbol{\sum w_i * x_i + b}$$

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

$$\mathbf{Y = f(\sum w_i * x_i + b)}$$

Types of Perceptron Models

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

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

Single Layer Perceptron Model:

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

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

Multi-Layered Perceptron Model:

Like a single-layer perceptron model, a multi-layer perceptron model also has the same model structure but has a greater number of hidden layers which executes in two stages as follows:

- **Forward Stage:** Activation functions start from the input layer in the forward stage and terminate on the output layer.
- **Backward Stage:** In the backward stage, weight and bias values are modified as per the model's requirement. In this stage, the error between actual output and demanded originated backward on the output layer and ended on the input layer.

A multi-layer perceptron model has greater processing power and can process linear and non-linear patterns. Further, it can also implement logic gates such as AND, OR, XOR, NAND, NOT, XNOR, NOR.

Advantages of Multi-Layer Perceptron:

- A multi-layered perceptron model can be used to solve complex non-linear problems.
- It works well with both small and large input data.
- It helps us to obtain quick predictions after the training.
- It helps to obtain the same accuracy ratio with large as well as small data.

Disadvantages of Multi-Layer Perceptron:

- In Multi-layer perceptron, computations are difficult and time-consuming.
- In multi-layer Perceptron, it is difficult to predict how much the dependent variable affects each independent variable.
- The model functioning depends on the quality of the training.

Characteristics of Perceptron

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

1.2.3 Logistic Regression

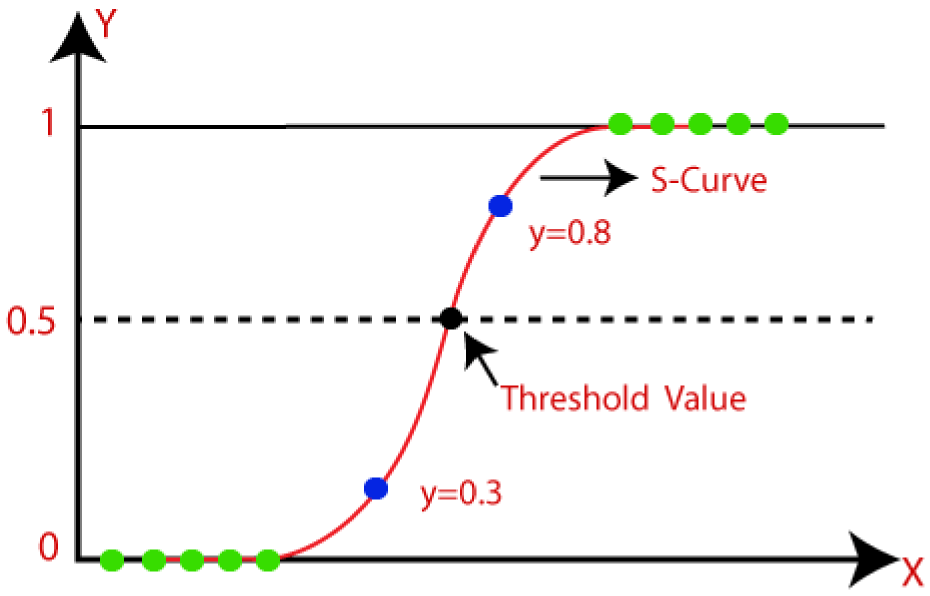
Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.

Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.

In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1). The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets. Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Logistic Regression Equation:

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

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

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

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

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

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

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

The above equation is the final equation for Logistic Regression.

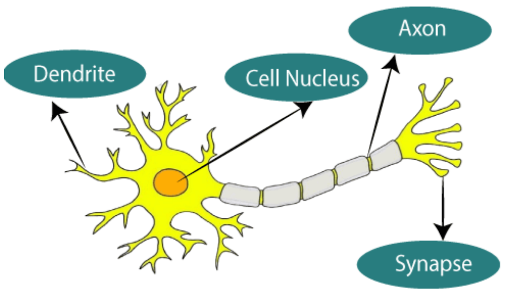
Type of Logistic Regression:

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

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

1.3 Intro to Neural Network

A neural network is **a method in artificial intelligence that teaches computers to process data in a way that is inspired by the human brain**. It is a type of machine learning process, called deep learning, that uses interconnected nodes or neurons in a layered structure that resembles the human brain.



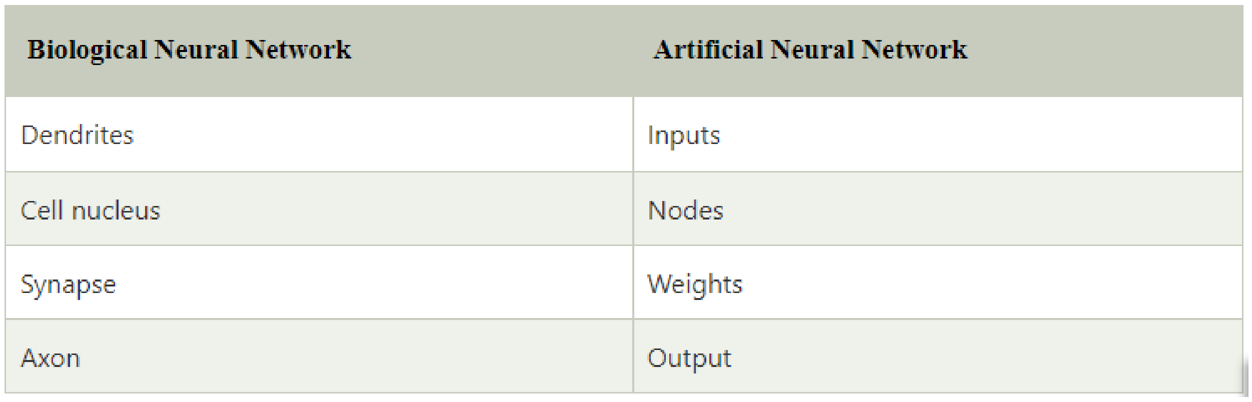
Different types of Neural Networks in Deep Learning

There are three important types of neural networks that form the basis for most pre-trained models in deep learning:

- Artificial Neural Networks (ANN)
- Convolution Neural Networks (CNN)
- Recurrent Neural Networks (RNN)

Artificial Neural Networks

The term "**Artificial Neural Network**" is derived from Biological neural networks that develop the structure of a human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks.



Biological Neural Network	Artificial Neural Network
Dendrites	Inputs
Cell nucleus	Nodes
Synapse	Weights
Axon	Output

Convolution neural network (CNN)

Convolution neural network contains a three-dimensional arrangement of neurons, instead of the standard two-dimensional array. The first layer is called a convolutional layer. Each neuron in the convolutional layer only processes the information from a small part of the visual field. Input features are taken in batch-wise like a filter. The network understands the images in parts and can compute these operations multiple times to complete the full image processing. Processing involves conversion of the image from RGB or HSI scale to grey-scale. Furthering the changes in the pixel value will help to detect the edges and images can be classified into different categories.

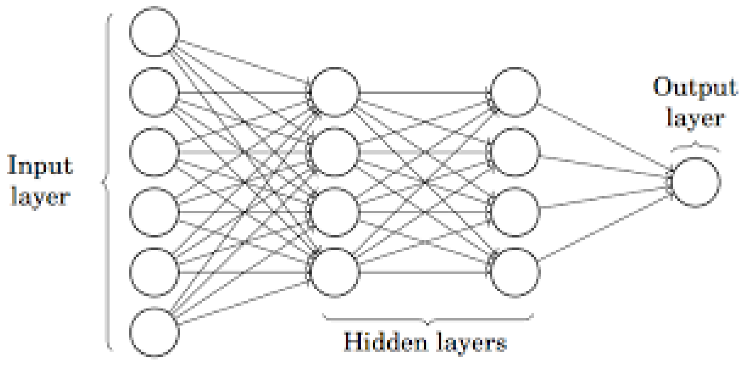
Recurrent Neural Network

(RNN) are a type of Neural Network where the output from previous step are fed as input to the current step. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words.

Architecture of Neural Network

A simple neural network consists of three components :

- Input layer
- Hidden layer
- Output layer



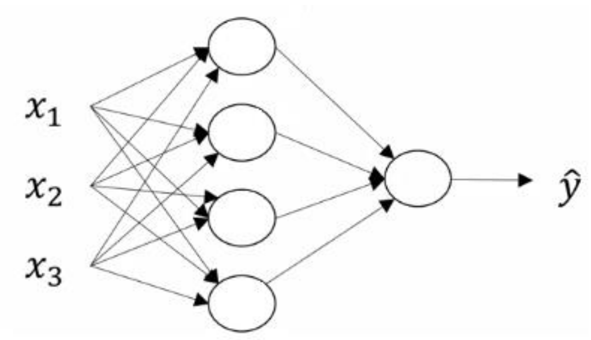
Input Layer: Also known as Input nodes are the inputs/information from the outside world is provided to the model to learn and derive conclusions from. Input nodes pass the information to the next layer i.e Hidden layer.

Hidden Layer: Hidden layer is the set of neurons where all the computations are performed on the input data. There can be any number of hidden layers in a neural network. The simplest network consists of a single hidden layer.

Output layer: The output layer is the output/conclusions of the model derived from all the computations performed. There can be single or multiple nodes in the output layer. If we have a binary classification problem the output node is 1 but in the case of multi-class classification, the output nodes can be more than 1.

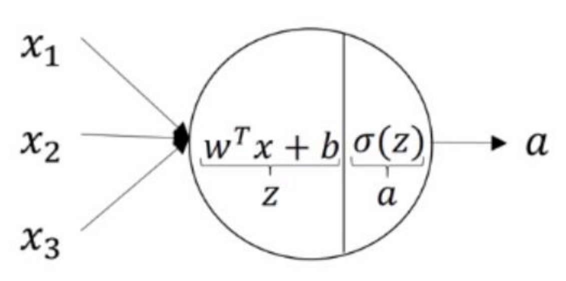
1.4 What a shallow network computes

"Shallow" neural networks is **a term used to describe NN that usually have only one hidden layer** as opposed to deep NN which have several hidden layers, often of various type. Shallow neural networks consist of only 1 or 2 hidden layers. Understanding a shallow neural network gives us an insight into what exactly is going on inside a deep neural network. In this post, let us see what is a shallow neural network and its working in a mathematical context. The figure below shows a shallow neural network with 1 hidden layer, 1 input layer and 1 output layer.



The Neuron

The neuron is the atomic unit of a neural network. Given an input, it provides the output and passes that output as an input to the subsequent layer. A neuron can be thought of as a combination of 2 parts:



- The first part computes the output **Z**, using the inputs and the weights.
- The second part performs the activation on **Z** to give out the final output **A** of the neuron.

The Hidden Layer

The hidden layer comprises of various neurons, each of which performs the above 2 calculations. The 4 neurons present in the hidden layer of our shallow neural network compute the following:

$$z_1^{[1]} = w_1^{[1]T} x + b_1^{[1]}, a_1^{[1]} = \sigma\left(z_1^{[1]}\right)$$
$$z_2^{[1]} = w_2^{[1]T} x + b_2^{[1]}, a_2^{[1]} = \sigma\left(z_2^{[1]}\right)$$
$$z_3^{[1]} = w_3^{[1]T} x + b_3^{[1]}, a_3^{[1]} = \sigma\left(z_3^{[1]}\right)$$
$$z_4^{[1]} = w_4^{[1]T} x + b_4^{[1]}, a_4^{[1]} = \sigma\left(z_4^{[1]}\right)$$

In the above equations,

1. The superscript number ***[i]*** denotes the layer number and the subscript number ***j*** denotes the neuron number in a particular layer.
2. ***X*** is the input vector consisting of 3 features.
3. ***W[i]j*** is the weight associated with neuron ***j*** present in the layer ***i***.
4. ***b[i]j*** is the bias associated with neuron ***j*** present in the layer ***i***.
5. ***Z[i]j*** is the intermediate output associated with neuron ***j*** present in the layer ***i***.
6. ***A[i]j*** is the final output associated with neuron ***j*** present in the layer ***i***.
7. ***Sigma*** is the sigmoid activation function. Mathematically it is defined as:

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

As we can see, the above 4 equations seem redundant. Therefore we will vectorize them as:

$$Z^{[1]} = X^{[1]T}X + b^{[1]}$$

$$A^{[1]} = \sigma\left(Z^{[1]}\right)$$

1.5 Training a Network

1. Pick a neural network architecture. This implies that you shall be pondering primarily upon the connectivity patterns of the neural network including some of the following aspects:
   - **Number of input nodes**: The way to identify number of input nodes is identify the number of features.
   - **Number of hidden layers**: The default is to use the single or one hidden layer. This is the most common practice.
   - **Number of nodes in each of the hidden layers**: In case of using multiple hidden layers, the best practice is to use same number of nodes in each hidden layer. In general practice, the number of hidden units is taken as comparable number to that of number of input nodes. That means one could take either the same number of hidden nodes as input nodes or maybe twice or thrice the number of input nodes.

- **Number of output nodes**: The way to identify number of output nodes is to identify the number of output classes you want the neural network to process.

2. Random Initialization of Weights: The weights are randomly intialized to value in between 0 and 1, or rather, very close to zero.
3. Implementation of forward propagation algorithm to calculate hypothesis function for a set on input vector for any of the hidden layer.
4. Implementation of cost function for optimizing parameter values. One may recall that cost function would help determine how well the neural network fits the training data.
5. Implementation of back propagation algorithm to compute the error vector related with each of the nodes.
6. Use gradient checking method to compare the gradient calculated using partial derivatives of cost function using back propagation and using numerical estimate of cost function gradient. The gradient checking method is used to validate if the implementation of backpropagation method is correct.
7. Use gradient descent or advanced optimization technique with back propagation to try and minimize the cost function as a function of parameters or weights.

1.6 Loss function

The loss function in a neural network **quantifies the difference between the expected outcome and the outcome produced by the machine learning model**. From the loss function, we can derive the gradients which are used to update the weights. The average over all losses constitutes the cost. Loss functions are divided into two categories

1. Regression loss
2. Classification Loss – Binary and Multi-class Classification

Regression Loss

1. Mean Squared Error (MSE)

Squared Error loss for each training example, also known as **L2 Loss**, is the square of the difference between the actual and the predicted values:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

2. Mean Square Logarithmic Error (MSLE)

It measures the ratio between actual and predicted using logarithmic values. It is a good choice to predict the continuous data.

$$MSLE = \frac{1}{n}\sum_{i=1}^{n}(\log(Y_i) - \log(\hat{Y}_i))^2$$

3. Mean Absolute Error (MAE)

Absolute Error for each training example is the distance between the predicted and the actual values, irrespective of the sign. Absolute Error is also known as the **L1 loss:**

$$MAE = \frac{\sum_{i=1}^{n} \mid y_i - \hat{y}_i \mid}{n}$$

Binary Classification Loss Functions

1. Binary Cross Entropy Loss

Entropy indicates disorder or uncertainty. It is measured for a random variable X with probability distribution p(X):

$$S = \begin{cases} -\int p(x).\log p(x).dx, & \text{if } x \text{ is continuous} \\ -\sum_{x} p(x).\log p(x), & \text{if } x \text{ is discrete} \end{cases}$$

Cross-entropy is the default loss function to use for binary classification problems. It is intended for use with binary classification where the target values are in the set {0, 1}.

2. Hinge Loss

An alternative to cross-entropy for binary classification problems is the hinge loss function, primarily developed for use with Support Vector Machine (SVM) models. It is intended for use with binary classification where the target values are in the set {-1, 1}.

3. Squared Hinge Loss

It is an extension of Hinge Loss. It is mainly used for categorical prediction or yes/no kind of decision problems.

Multi-Class Classification Loss Functions

Multi-Class classification are those predictive modeling problems where examples are assigned one of more than two classes.

1. Multiclass Cross Entropy Loss

Cross-entropy is the default loss function to use for multi-class classification problems. In this case, it is intended for use with multi-class classification where the target values are in the set {0, 1, 3, …, n}, where each class is assigned a unique integer value. It ia best loss function for text classification.

2. Sparse Multiclass Cross-Entropy Loss

Sparse cross-entropy addresses the same performance as cross-entropy calculation of error and it is mainly used to handle large amount of data.

1.7 Backpropagation and Stochastic Gradient Descent

Feedforward Neural Network :

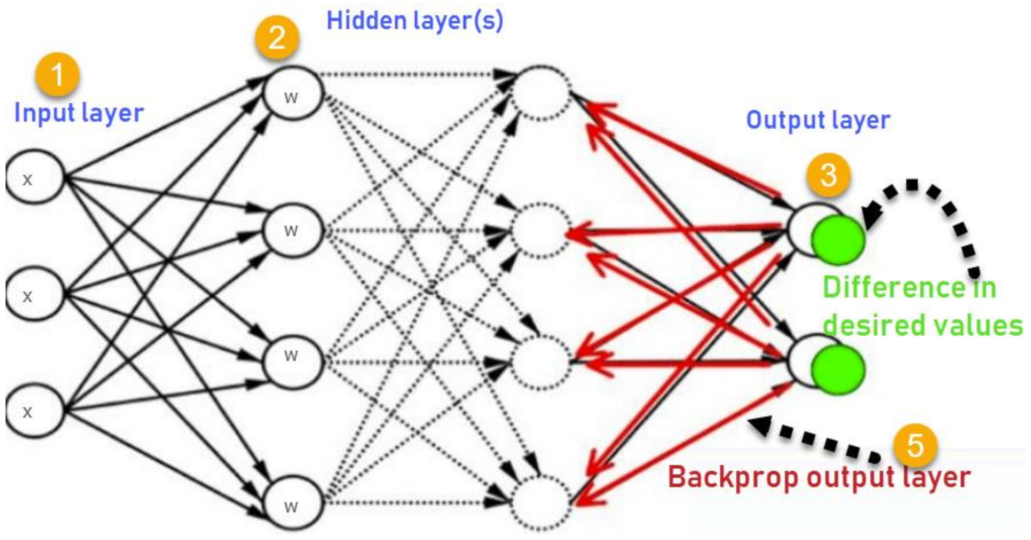
A feedforward neural network is an artificial neural network where the nodes never form a cycle. This kind of neural network has an input layer, hidden layers, and an output layer. It is the first and simplest type of artificial neural network.

Backpropagation

Backpropagation is the essence of neural network training. It is the feedforward network with feedback. 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.

The Back propagation algorithm in neural network computes the gradient of the loss function for a single weight by the chain rule. It efficiently computes one layer at a time, unlike a native direct computation. It computes the gradient, but it does not define how the gradient is used. It generalizes the computation in the delta rule.



Backpropagation Algorithm:

1. Inputs X, arrive through the preconnected path
2. Input is modelled using real weights W. The weights are usually randomly selected.
3. Calculate the output for every neuron from the input layer, to the hidden layers, to the output layer.
4. Calculate the error in the outputs

   Error = Actual Output – Desired Output

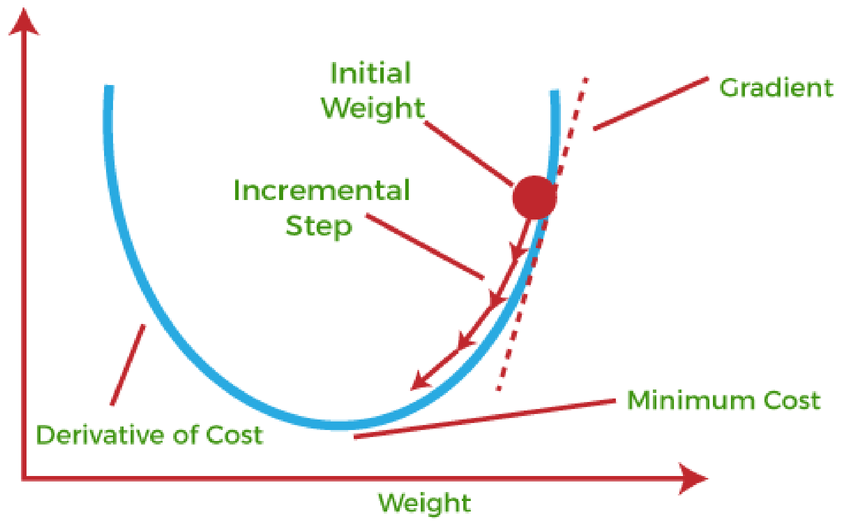
5. Travel back from the output layer to the hidden layer to adjust the weights such that the error is decreased.

Gradient Descent:

Gradient descent is an optimization algorithm which is commonly-used to train machine learning models and neural networks. Optimization algorithm refers to the task of minimizing/maximizing an objective function f(x) parameterized by x. Similarly, in machine learning, optimization is the task of minimizing the cost function parameterized by the model's parameters. The main objective of gradient descent is to minimize the convex function using iteration of parameter updates. It helps in finding the local minimum of a function.

The best way to define the local minimum or local maximum of a function using gradient descent is as follows:

- If we move towards a negative gradient or away from the gradient of the function at the current point, it will give the **local minimum** of that function.
- Whenever we move towards a positive gradient or towards the gradient of the function at the current point, we will get the **local maximum** of that function.



Types of Gradient Descent:

Typically, there are three types of Gradient Descent:

1. Batch Gradient Descent
2. Stochastic Gradient Descent
3. Mini-batch Gradient Descent

Stochastic Gradient Descent (SGD):

The word ‘stochastic‘ means a system or a process that is 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 totalnumber 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, but the problem arises when our datasets get big. Suppose, you have a million samples in your dataset, so if you use a typical

Gradient Descent optimization technique, you will have to use all of the one million samples for completing one iteration while performing the Gradient Descent, and it has to be done for every iteration until the minima is reached. Hence, it becomes computationally very expensive to perform.

Advantages

- Good Efficiency.
- Ease of implementation

Disadvantages

- It requires a number of hyperparameters such as the regularization parameter and the number of iterations.
- It is sensitive to feature scaling.

Questions to Revise:

Part A:

1. Define Machine Learning. List out its types.
2. List out the applications of machine learning.
3. Differentiate supervised and unsupervised learning algorithm.
4. List out any 4 linear model algorithms for classification.
5. What a shallow network computes?
6. List out the types of logistic regression.
7. Give the relation diagram of AI, ML and DL.
8. Define stochastic gradient descent and give its advantages.
9. Write the universal approximation theorem for neural networks.
10. Expand SVM. List out its advantages and disadvantages.
11. Differentiate linear and logistic regression.
12. What is meant by support vectors? What is its significance?
13. Briefly explain the basic components of a neural network.
14. Give a simple diagrammatic representation of a neural network.
15. What is meant by batch? What are the batch size for different gradient descent types?

Part B:

1. Define Machine learning and explain in detail about its types.
2. Give a brief explanation about how supervised and unsupervised learning algorithm

works with examples.

3. Explain in detail about any 2 linear models.
4. Explain in detail about the following:
    a) SVM
    b) Logistic regression
5. Explain in detail about perceptron with neat sketches.
6. What is meant by artificial neural network? Explain its basic types?
7. Explain in detail about “how to train a neural network”.
8. Define back propagation. Formulate an algorithm to design a backpropagation network. List out its advantages and disadvantages.
9. a) What is the significance of back propagation networks with feed forward networks?

    b) How to compute the error term?

    c) Explain in detail about the different types of loss functions.
10. Define gradient descent and list out its types. Explain in detail about Stochastic gradient descent.