UNIT-5: Machine Learning (9 Hrs)

Syllabus

Unit V: Machine Learning (9 Hrs.)

5.1. Introduction to Machine Learning, Concepts of Learning, Supervised, Unsupervised and Reinforcement Learning
5.2. Statistical-based Learning: Naive Bayes Model
5.3. Learning by Genetic Algorithm
5.4. Learning with Neural Networks: Introduction, Biological Neural Networks Vs. Artificial Neural Networks (ANN), Mathematical Model of ANN, Types of ANN: Feed-forward, Recurrent, Single Layered, Multi-Layered, Application of Artificial Neural Networks, Learning by Training ANN, Supervised vs. Unsupervised Learning, Hebbian Learning, Perceptron Learning, Back-propagation Learning

---

Introduction to Machine Learning

Machine learning refers to the study of computer systems that learn and adapt automatically from experience, without being explicitly programmed. With machine learning models, computer scientists can "train" a machine by feeding it large amounts of data. The machine follows a set of rules—called an algorithm—to analyse and draw inferences from the data. The more data the machine parses, the better it can become at performing a task or making a decision.

Hence,

- Machine learning is an area of artificial intelligence (AI) with a concept that, how a computer program can learn and adapt to new data without human intervention.
- A complex algorithm or source code is built into a computer that allows for the machine to identify data and build predictions around the data that it identifies.
- Machine learning is useful in parsing the immense amount of information that is consistently and readily available in the world to assist in decision making.
- Machine learning can be applied in a variety of areas, such as in investing, advertising, lending, organizing news, fraud detection, and more.

Example Like Music streaming service Spotify learns your music preferences to offer you new suggestions. Each time you indicate that you like a song by listening through to the end or adding it to your library, the service updates its algorithms to feed you more accurate recommendations. Netflix and Amazon use similar machine learning algorithms to offer personalized recommendations.

How machine learning works

Machine learning is a form of artificial intelligence (AI) that teaches computers to think in a similar way to how humans do: Learning and improving upon past experiences. It works by exploring data and identifying patterns, and involves minimal human intervention.

The early stages of machine learning (ML) saw experiments involving theories of computers recognizing patterns in data and learning from them. Today, after building upon those foundational experiments, machine learning is more complex.

The Machine Learning process starts with inputting training data into the selected algorithm. Training data being known or unknown data to develop the final Machine Learning algorithm. The type of training data input does impact the algorithm, and that concept will be covered further momentarily.

New input data is fed into the machine learning algorithm to test whether the algorithm works correctly. The prediction and results are then checked against each other. If the prediction and results don't match, the algorithm is re-trained multiple times until the data scientist gets the desired outcome. This enables the machine learning algorithm to continually learn on its own and produce the optimal answer, gradually increasing in accuracy over time.

There are three key aspects of machine learning which are the following:

- **Task:** Task can be related to prediction problems such as classification, regression, clustering, etc
- **Experience:** Experience represents the historical dataset
- **Performance:** The goal is to perform better in the prediction tasks based on the past datasets. Different performance measures exist for different kinds of machine learning problems.

Machine learning is different from traditional programming in the way that the Traditional programming languages typically take data and (business or solution) rules as input and apply the rules to the data in order to come up with answers as output. For example, a traditionally programmed sales analysis application will read in sales data along with (business or sales) rules, and then apply those rules to the data and output the key trends, analytics or insights.

On the other hand, in machine learning paradigm the data and answers (or labels) go in as input and the learned rules (models) come out as output. Machine learning paradigm is uniquely valuable because it lets computer to learn new rules in complex and high dimensional space, a space harder to comprehend by humans.

Fig: Difference between Machine learning and traditional programming

<u>General Machine Learning agent</u>

A learning agent in AI is the type of agent which can learn from its past experiences or it has learning capabilities. It starts to act with basic knowledge and then able to act and adapt automatically through learning.

Or

A **learning agent** is a tool in AI that is capable of learning from its experiences. It starts with some basic knowledge and is then able to act and adapt autonomously, through learning, to improve its own performance. Unlike intelligent agents that act on information provided by a programmer, learning agents are able to perform tasks, analyse performance, and look for new ways to improve on which is shown in figure below.

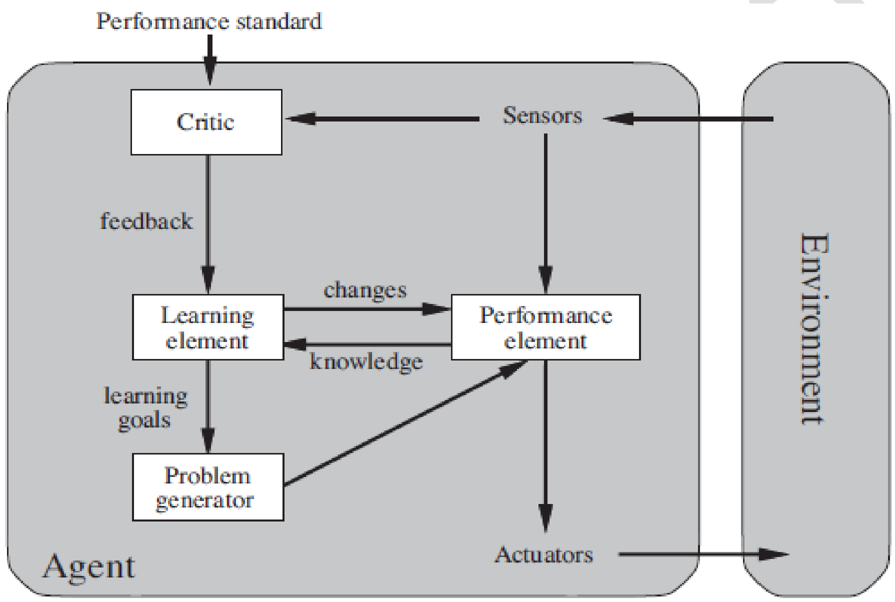


Figure: A general learning agent.

As shown in above figure a learning agent has mainly four conceptual components, which are:

<u>Critic:</u>

It is the one who compares sensor's input to specifying the effects of agent's action to the environment with the preference standard and generates feedback for leaning element.

Learning element:

- This component is responsible to learn from difference between preference standard and feedback from critics.
- According to current percept it is supposed to understand the expected behaviour and enhance its standards.

Performance element:

Based on the current percept received from sensor and the input obtained by the learning element performance element is responsible to act upon the external environment.

Problem Generator:

Based on the new goal learn by the learning agent, problem generator suggests new or alternative action which will leads to new and instructive understanding.

Examples of ML in use.

- **Prediction** — Machine learning model predictions allow businesses to make highly accurate guesses as to the likely outcomes of a question based on historical data, which can be about all kinds of things – customer churn likelihood, possible fraudulent activity, and more.
- **Image recognition** — Machine learning can be used for face detection in an image as well. There is a separate category for each person in a database of several people.
- **Speech Recognition —** It is the translation of spoken words into the text. It is used in voice searches and more. Voice user interfaces include voice dialling, call routing, and appliance control. It can also be used as simple data entry and the preparation of structured documents.
- **Medical diagnoses —** ML is trained to recognize cancerous tissues.

Applications of Machine Learning

- **Web search:** ranking page based on what you are most likely to click on.
- **Computational biology:** rational design drugs in the computer based on past experiments.
- **Finance:** decide who to send what credit card offers to. Evaluation of risk on credit offers. How to decide where to invest money.
- **E-commerce:** Predicting customer churn. Whether or not a transaction is fraudulent.
- **Space exploration:** space probes and radio astronomy.
- **Robotics:** how to handle uncertainty in new environments. Autonomous. Self-driving car.
- **Information extraction:** Ask questions over databases across the web.
- **Social networks:** Data on relationships and preferences. Machine learning to extract value from data.
- **Debugging:** Use in computer science problems like debugging.

Artificial intelligence V/S machine learning

Artificial intelligence is the capability of a computer system to mimic human cognitive functions such as learning and problem-solving. Through AI, a computer system uses math and logic to simulate the reasoning that people use to learn from new information and make decisions.

Machine learning is an application of AI. It’s the process of using mathematical models of data to help a computer learn without direct instruction. This enables a computer system to continue learning and improving on its own, based on experience.

Artificial Intelligence	**Machine learning**

Artificial intelligence is a technology which enables a machine to simulate human behaviour.	Machine learning is a subset of AI which allows a machine to automatically learn from past data without programming explicitly.
The goal of AI is to make a smart computer system like humans to solve complex problems.	The goal of ML is to allow machines to learn from data so that they can give accurate output.
In AI, we make intelligent systems to perform any task like a human.	In ML, we teach machines with data to perform a particular task and give an accurate result.
AI is working to create an intelligent system which can perform various complex tasks.	Machine learning is working to create machines that can perform only those specific tasks for which they are trained.
AI system is concerned about maximizing the chances of success.	Machine learning is mainly concerned about accuracy and patterns.
The main applications of AI are Siri, customer support using catboats, Expert System, Online game playing, intelligent humanoid robot, etc.	The main applications of machine learning are Online recommender system, Google search algorithms, Facebook auto friend tagging suggestions, etc.

Supervised learning

Supervised learning is a form of machine learning in which machines are taught using well-labelled training data and then predict results using that data. The labelled data ensures that certain input data have the right output already marked.

The training data supplied to the machines in supervised learning work like a supervisor who trains the machines to accurately forecast the results. It uses the same principle that a pupil knows through the teacher's supervision.

In supervised learning, learning data comes with description, labels, targets or desired outputs and the objective is to find a general rule that maps inputs to outputs. This kind of learning data is called labelled data. The learned rule is then used to label new data with unknown outputs. Supervised learning involves building a machine learning model that is based on labelled samples.

For example, if we build a system to estimate the price of a plot of land or a house based on various features, such as size, location, and so on, we first need to create a database and label it. We need to teach the algorithm what features correspond to what prices. Based on this data, the algorithm will learn how to calculate the price of real estate using the values of the input features.

Supervised learning deals with learning a function from available training data. Here, a learning algorithm analyses the training data and produces a derived function that can be used for mapping new examples. There are many supervised learning algorithms such as Logistic Regression, Neural networks, Support Vector Machines (SVMs), and Naive Bayes classifiers.

Common examples of supervised learning include classifying e-mails into spam and not-spam categories, labelling webpages based on their content, and voice recognition.

Similarly, Supervised learning is commonly used in real world applications, such as face and speech recognition, products or movie recommendations, and sales forecasting. Supervised learning can be further classified into two types: - Regression and Classification.

Classification

Classification is defined as the process of recognition, understanding, and grouping of objects and ideas into pre-set categories also known as "sub-populations." With the help of these pre-categorized training datasets, classification in machine learning programs leverage a wide range of algorithms to classify future datasets into respective and relevant categories.

Classification algorithms used in machine learning utilize input training data for the purpose of predicting the likelihood or probability that the data that follows will fall into one of the predetermined categories. One of the most common applications of classification is for filtering emails into "spam" or "non-spam", as used by today's top email service providers.

In short, classification is a form of "pattern recognition,". Here, classification algorithms applied to the training data find the same pattern (similar number sequences, words or sentiments, and the like) in future data sets.

As we already know that Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modelling is the task of approximating a mapping function (f) from input variables (x) to discrete output variables (y).

y=f(x), where y = categorical output

The output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc. Since the Classification algorithm is a Supervised learning technique, hence it takes labelled input data, which means it contains input with the corresponding output.

There are two types of Classifications:

- Binary Classifier: If the classification problem has only two possible outcomes, then it is called as Binary Classifier.
  Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.
- Multi-class Classifier: If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.
  Example: Classifications of types of crops, Classification of types of music.

Regression

Regression in machine learning refers to a type of supervised learning where the goal is to predict a continuous numerical value based on input data; that is Regression is a technique in machine learning for predicting continuous outcomes based on input data. It is one of the most common tasks in machine learning and is used in various applications, such as predicting prices, estimating risks, or forecasting sales.

Regression is a field of study in statistics which forms a key part of forecast models in machine learning. It is a method for understanding the relationship between independent variables or features and a dependent variable or outcome. Outcomes can then be predicted once the relationship between independent and dependent variables has been estimated.

- Independent variables (features): These are the input variables used to predict the output. They are also called predictors or explanatory variables.
- Dependent variable (target): This is the output variable that you are trying to predict.

In regression, a model is trained on labelled data, where each data point consists of input features (independent variables) and a corresponding output value (dependent variable). The goal is to learn a mapping from the input features to the output that can be used to make predictions on new, unseen data.

Types of Regression Models:

- **Linear Regression**: Predicts the dependent variable as a linear combination of the independent variables. The model assumes a linear relationship between the input and output.
- **Polynomial Regression**: An extension of linear regression, where the relationship between the independent variable and the dependent variable is modeled as an nth-degree polynomial.
- **Ridge and Lasso Regression**: Ridge and Lasso Regression are both regularized versions of linear regression that aim to prevent overfitting by adding a penalty to the loss function. Ridge regression uses L2 regularization, while Lasso uses L1 regularization.
- **Logistic Regression**: Despite its name, it's used for binary classification rather than regression. However, it can be extended to multi-class problems.
- **Support Vector Regression (SVR)**: A type of regression that uses the principles of Support Vector Machines (SVM) i.e. classification tasks, but adapted for continuous outputs.
- **Decision Trees and Random Forests**: These models can be used for regression tasks by predicting the average output value for the data points in a given region of the feature space.

Unsupervised learning

Unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things.

It can be defined as the type of machine learning in which models are trained using unlabelled dataset and are allowed to act on that data without any supervision. Here, the data provided to the model does not include labels or explicit output values. The model must learn from the data itself without guidance on what the "correct" answers are.

Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.

Some applications of unsupervised machine learning techniques include:

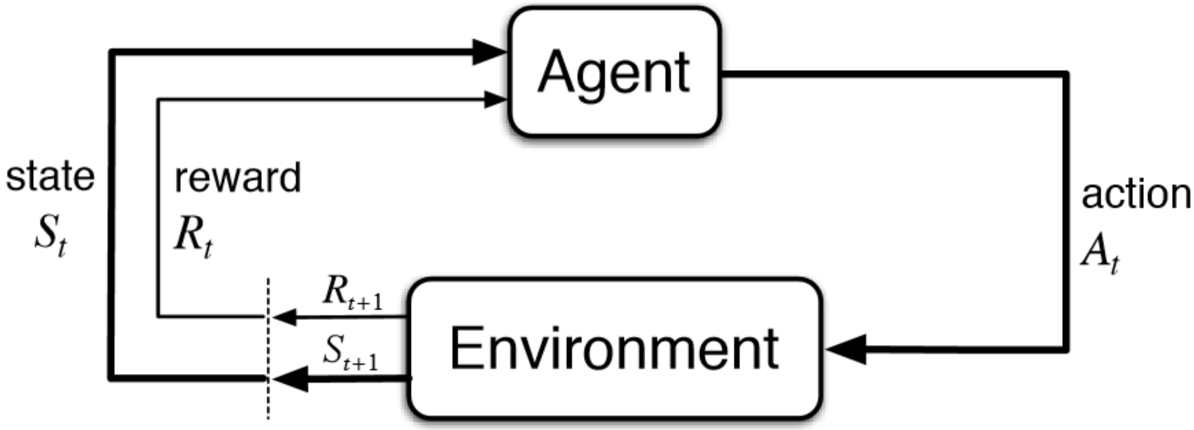
- **Clustering** allows you to automatically split the dataset into groups according to similarity.
- **Anomaly detection** can automatically discover unusual data points in your dataset. This is useful in pinpointing fraudulent transactions, discovering faulty pieces of hardware, or identifying an outlier caused by a human error during data entry.
- **Association mining** identifies sets of items that frequently occur together in your dataset. Retailers often use it for basket analysis, because it allows analysts to discover goods often purchased at the same time and develop more effective marketing and merchandising strategies.
- **Customer Segmentation**: Grouping customers based on purchasing behavior, demographics, or other characteristics for targeted marketing.
- **Data Compression**: Reducing the dimensionality of data for efficient storage and transmission.
- **Image and Speech Recognition**: Extracting features from raw data to identify objects in images or recognize spoken words.

Reinforcement learning

Reinforcement Learning (RL) is a type of machine learning technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences.

As Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty.

The figure below illustrates the action-reward feedback loop of a generic RL model.



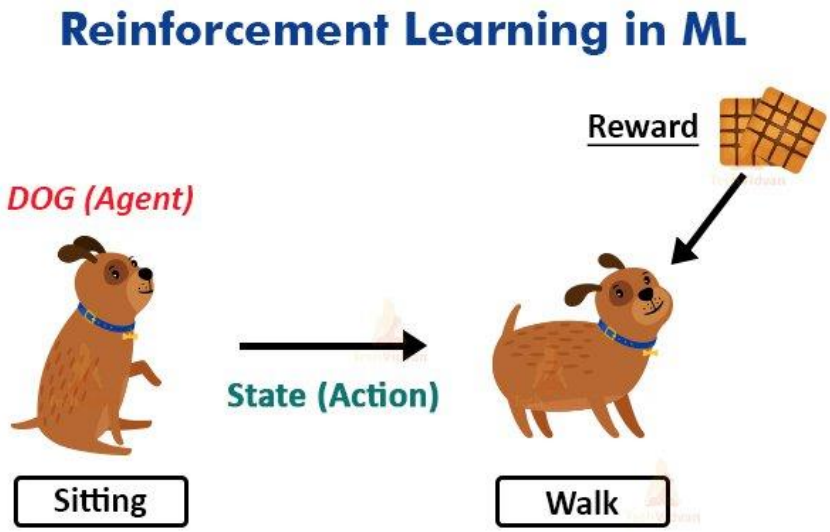
In Reinforcement Learning, the agent learns automatically using feedbacks without any labelled data, unlike supervised learning.

Since there is no labelled data, so the agent is bound to learn by its experience only. It solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc.

The primary goal of an agent in reinforcement learning is to improve the performance by getting the maximum positive rewards. The agent learns with the process of hit and trial, and based on the experience, it learns to perform the task in a better way.

Hence, we can say that "Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that." How a Robotic dog learns the movement of his arms is an example of Reinforcement learning.

Example of Reinforcement Learning



Let’s say you have a dog and you are trying to train your dog to sit. You would give certain instructions to the dog to try to make it learn. If the dog executes the instruction perfectly, it would get a biscuit as a reward. If not, it would not get anything. The dog learns from this after some tries that it would get a biscuit if it sits.
This is what the gist of reinforcement learning is. The reward here is the feedback received by the dog for sitting. This algorithm has various applications in real life.

Statistical-based Learning: Naive Bayes Model

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, which can be described as:

- **Naïve:** It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- **Bayes:** It is called Bayes because it depends on the principle of Bayes' Theorem

Bayes' Theorem:

Bayes' theorem is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

The formula for Bayes' theorem is given as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Where,

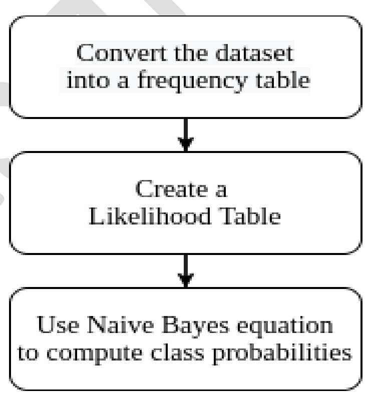
- P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.
- P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.
- P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
- P(B) is Marginal Probability: Probability of Evidence.

Working of Naïve Bayes' Classifier:

Working of Naïve Bayes' Classifier can be understood with the help of the below example: Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions.

So to solve this problem, we need to follow the below steps:

- Convert the given dataset into frequency tables.
- Generate Likelihood table by finding the probabilities of given features.
- Now, use Bayes theorem to calculate the posterior probability.



Problem: If the weather is sunny, then the Player should play or not?
Solution: To solve this, first consider the below dataset:

Sr no	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Frequency table for the Weather Conditions:

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	4

Likelihood table weather condition:

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

Applying Bayes'theorem:
P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)

P(Sunny|Yes)= 3/10= 0.3
P(Sunny)= 0.35
P(Yes)=0.71
So P(Yes|Sunny) = 0.3*0.71/0.35= 0.60

P(No|Sunny)= P(Sunny|No)*P(No)/P(Sunny)

P(Sunny|NO)= 2/4=0.5
P(No)= 0.29
P(Sunny)= 0.35
So P(No|Sunny)= 0.5*0.29/0.35 = 0.41

So as we can see from the above calculation that **P(Yes|Sunny)>P(No|Sunny)**

Hence on a Sunny day, Player can play the game.

Advantages of Naïve Bayes Classifier:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for text classification problems.

Disadvantages of Naïve Bayes Classifier:

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.
- Applications of Naïve Bayes Classifier:
- It is used for Credit Scoring.
- It is used in medical data classification.
- It can be used in real-time predictions because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as Spam filtering and Sentiment analysis.

Learning by Genetic Algorithm

Charles Darwin has formulated the fundamental principle of natural selection as the main evolutionary tool. He put forward his ideas without the knowledge of basic hereditary principles. In 1865, Gregor Mendel discovered these hereditary principles by the experiments he carried out on peas. After Mendel's work genetics was developed. Morgan experimentally found that chromosomes were the carriers of hereditary information and chat genes representing the hereditary factors were lined up on chromosomes. Combination of Darwin's and Mendel's ideas lead to the modern evolutionary theory.

Introduction

A genetic algorithm is a search heuristic that is inspired by Charles Darwin's theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.

A genetic algorithm is a search-based algorithm used for solving optimization problems in machine learning. It is used to solve complicated problems with a greater number of variables & possible outcomes/solutions. The combinations of different solutions are passed through the Darwinian based algorithm to find the best solutions. The poorer solutions are then replaced with the offspring of good solutions.

It all works on the Darwinian theory, where only the fittest individuals are chosen for reproduction. The various solutions are considered the elements of the population, and only the fittest solutions are allowed to reproduce (to create better solutions). Genetic algorithms help in optimizing the solutions to any particular problem.

It is a heuristic search algorithm used to solve search and optimization problems. These algorithms have better intelligence than random search algorithms because they use historical data to take the search to the best performing region within the solution space.

This algorithm is a subset of evolutionary algorithms, which are used in computation. Genetic algorithms employ the concept of genetics and natural selection to provide solutions to problems.

Basic terminologies in genetic algorithm:

- **Population:** Population is the subset of all possible or probable solutions(A collection of chromosomes) which can solve the given problem.

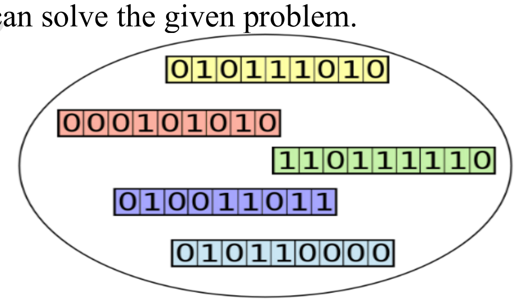


Fig: Population

- **Chromosomes:**

  A chromosome is one of the solutions in the population for the given problem, and the collection of gene generate a chromosome.

- **Gene:**
  A chromosome is divided into a different gene, or it is an element of the chromosome.

- **Allele:**
  Allele is the value provided to the gene within a particular chromosome.
- **Fitness Function:**
  The fitness function is used to determine the individual's fitness level in the population. It means the ability of an individual to compete with other individuals. In every iteration, individuals are evaluated based on their fitness function.
- **Genetic Operators:**
  In a genetic algorithm, the best individual mate to regenerate offspring better than parents. Here genetic operators play a role in changing the genetic composition of the next generation.

How Genetic Algorithm Works

The genetic algorithm works on the evolutionary generational cycle to generate high-quality solutions. These algorithms use different operations that either enhance or replace the population to give an improved fit solution. It basically involves five phases to solve the complex optimization problems, which are given as below:

1) Initialization
2) Fitness Assignment
3) Selection
4) Reproduction
5) Termination

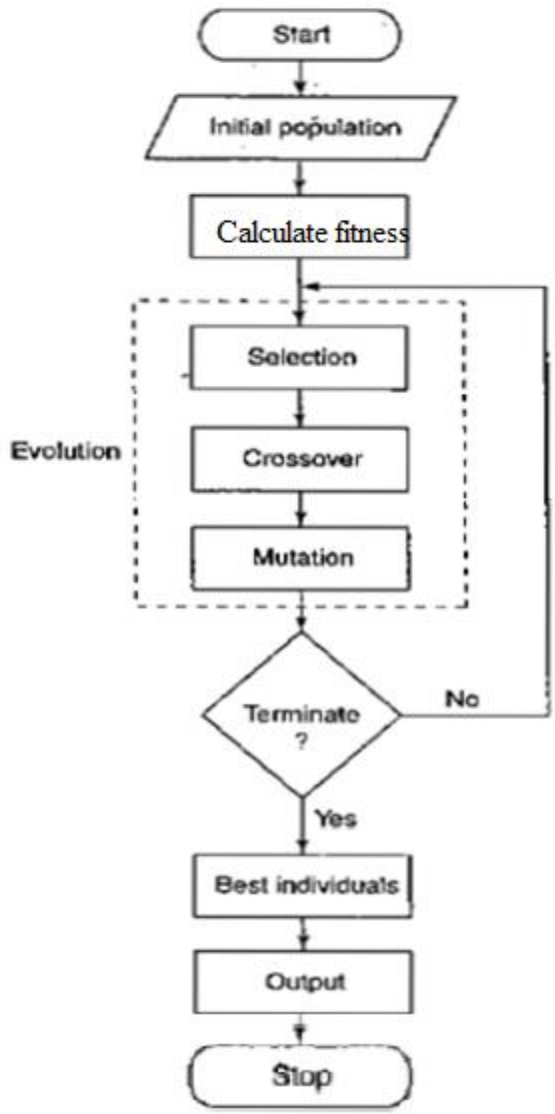


Fig: - Flowchart for generic algorithm.

1. Initialization

The process of a genetic algorithm starts by generating the set of individuals, which is called population. Here each individual is the solution for the given problem. An individual contains or is characterized by a set of parameters called Genes. Genes are combined into a string and generate chromosomes, which is the solution to the problem. One of the most popular techniques for initialization is the use of random binary strings.

2. Fitness Assignment

Fitness function is used to determine how fit an individual is? It means the ability of an individual to compete with other individuals. In every iteration, individuals are evaluated based on their fitness function. The fitness function provides a fitness score to each individual. This score further determines the probability of being selected for reproduction. The high the fitness score, the more chances of getting selected for reproduction.

Example:

Consider 8-bit chromosomes with the following properties:

Table:- Fitness value for corresponding chromosomes

Chromosome	Fitness
A: 00000110	2
B: 11101110	6
C: 00100000	1
D: 00110100	3

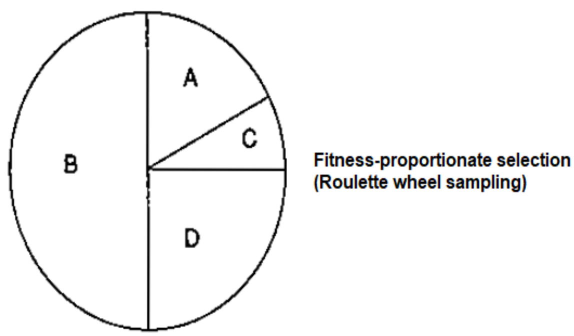


Figure (a): Roulette wheel sampling for fitness-propionate selection.

Now,

1. Fitness function f(x) = number of 1 bit in chromosome;
2. population size N = 4;
3. Crossover probability p = 0.7;
4. Mutation probability Pm = 0.001;

∴ Average fitness of population= 12/4 = 3.0.

3. Selection

Selection is the process of choosing two parents from the population for crossing. The selection phase involves the selection of individuals for the reproduction of offspring. All the selected individuals are then arranged in a pair of two to increase reproduction. Then these individuals transfer their genes to the next generation.

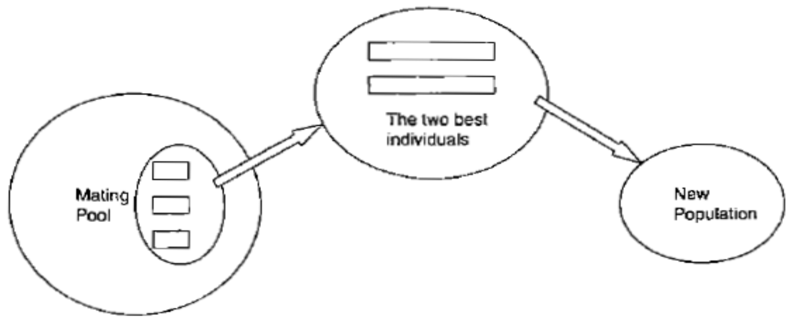


Figure: - Selection.

There are three types of Selection methods available, which are:

1) Roulette wheel selection

   Roulette Wheel Selection, also known as fitness proportionate selection, is a probabilistic selection method. It assigns a probability of selection to each individual based on their fitness, with higher fitness individuals having a greater chance of being selected.

2) Tournament selection

   Tournament Selection is a method that involves running several "tournaments" among a few individuals randomly chosen from the population. The winner of each tournament (the individual with the highest fitness) is selected for the next generation.

3) Rank-based selection

   Rank-Based Selection assigns selection probabilities based on the rank of an individual's fitness, rather than the fitness value itself. The population is sorted based on fitness, and selection probabilities are assigned according to ranks.

4. Reproduction

After the selection process, the creation of a child occurs in the reproduction step. In this step, the genetic algorithm uses two variation operators that are applied to the parent population. The two operators involved in the reproduction phase are given below:

Crossover:

Crossover is the process of taking two parent solutions and producing from them a child. After the selection (reproduction) process, the population is enriched with better individuals. Reproduction makes clones of good strings but does not create new ones. Crossover operator is applied to the mating pool with the hope that it creates a better offspring.

Crossover is a recombination operator that proceeds in three steps:

1. The reproduction operator selects at random a pair of two individual strings for the mating.
2. A cross site is selected at random along the string length.
3. Finally, the position values are swapped between the two strings following the cross site.

That is the simplest way how to do that is to choose randomly some crossover point and copy everything before this point from the first parent and then copy everything after the crossover point from the other parent. The various crossover techniques are discussed in the following subsections.

Types of crossover styles available:

- Single point crossover
- Two-point crossover
- Livery crossover
- Inheritable Algorithms crossover

I. Single-Point Crossover

The traditional genetic algorithm uses single-point crossover, where the two mating chromosomes are cut once at corresponding points and the sections after the cuts exchanged.

Here, a cross site or crossover point is selected randomly along the length of the mated strings and bits next to the cross sites are exchanged. If appropriate site is chosen, better children can be obtained by combining good parents, else it severely hampers string quality.

Figure below illustrates single point crossover and it can be observed that the bits next to the crossover point are exchanged to produce children. The crossover point can be chosen randomly.

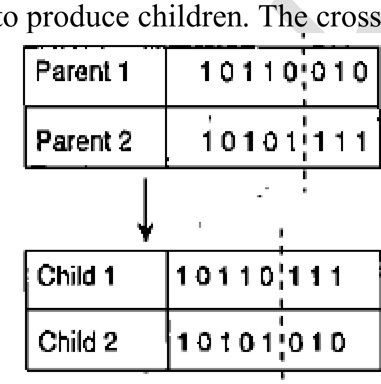


Figure: Single-point crossover.

The crossover plays a most significant role in the reproduction phase of the genetic algorithm. In this process, a crossover point is selected at random within the genes. Then the crossover operator swaps genetic information of two parents from the current generation to produce a new individual representing the offspring.

The genes of parents are exchanged among themselves until the crossover point is met. These newly generated offspring are added to the population. This process is also called or crossover.

2.Two-Point Crossover

In two-point crossover, two crossover points are chosen on the parent chromosomes. The segments between these points are swapped between the parents to produce the offspring.

- Two parent chromosomes are selected.
- Two random crossover points are selected along the length of the chromosomes.
- The genes between the two crossover points are swapped between the two parents to create the offspring.

Example:
Parent 1: 101|110|01
Parent 2: 010|001|11
Offspring 1: 10100101
Offspring 2: 01011011

3. Livery Crossover (Likely referring to Uniform Crossover)

In Uniform Crossover, each gene in the offspring is independently chosen from either parent with equal probability. This method does not rely on fixed crossover points but instead uses a random decision for each gene.

- For each gene in the chromosome, a random decision is made to take the gene from either Parent 1 or Parent 2.
- This process is repeated for all genes to create the offspring.

Example:
Parent 1: 10111001
Parent 2: 01000111
Offspring 1: 11101001
Offspring 2: 00010111

In more detail…

Crossover Process:

For each gene (or bit) position in the offspring chromosome, a random decision is made to choose the gene from either Parent 1 or Parent 2.

Example:
Consider the parents:
Parent 1: 10111001
Parent 2: 01000111

For each gene position (from left to right), you randomly decide which parent to take the gene from.

Suppose the random decisions for each gene are as follows:

- Position 1: Choose from Parent 1
- Position 2: Choose from Parent 1
- Position 3: Choose from Parent 2
- Position 4: Choose from Parent 1
- Position 5: Choose from Parent 2
- Position 6: Choose from Parent 2
- Position 7: Choose from Parent 1
- Position 8: Choose from Parent 2

Offspring 1 is generated by selecting:

- From Parent 1: 1 1 0 1 1 1 0 1
- From Parent 2: 0 0 1 0 0 0 1 1

  Offspring 1: 10100101

Similarly, using different random choices, you might generate another offspring:
Suppose for Offspring 2, the random decisions were:

- Position 1: Choose from Parent 2
- Position 2: Choose from Parent 1
- Position 3: Choose from Parent 2
- Position 4: Choose from Parent 1
- Position 5: Choose from Parent 1
- Position 6: Choose from Parent 1
- Position 7: Choose from Parent 2
- Position 8: Choose from Parent 2

Offspring 2: 01011111

4. Inheritable Algorithms Crossover

This term is less common and might refer to Inheritable Genetic Algorithms (IGAs), where crossover and mutation operations are designed to allow certain traits or genes to be more easily inherited by offspring. The crossover methods in IGAs are often adapted to maintain the integrity of these inheritable traits across generations.

Objectives

- Preservation of Traits: In IGAs, certain genes or traits are considered important and need to be preserved across generations. This preservation helps in maintaining high-quality solutions and ensures that beneficial features are not lost during the evolution process.
- Controlled Diversity: While preserving key traits, IGAs also need to maintain genetic diversity to explore new areas of the solution space. The challenge is to balance preservation with exploration.

How It Works:

In IGAs, the crossover operation is often guided by specific rules or heuristics to ensure that key traits (genes or gene sequences) are passed on to offspring without disruption. The crossover can be tailored to preserve these traits while still allowing genetic diversity.

Mutation:

After crossover, the strings are subjected to mutation. Mutation plays the role of recovering the lost genetic materials as well as for randomly distributing generic information. It is an insurance policy against the irreversible loss of genetic material.

The mutation operator inserts random genes in the offspring (new child) to maintain the diversity in the population. It can be done by flipping some bits in the chromosomes.

Mutation helps in solving the issue of premature convergence and enhances diversification. The below image shows the mutation process: Types of mutation styles available,

- Flip bit mutation
- Gaussian mutation
- Exchange/Swap mutation

1.Flipping

Flipping of a bit involves changing 0 to 1 and 1 to 0 based on a mutation chromosome generated. Figure below explains mutation flipping concept.

Parent	1 0 1 1 0 1 0 1
Mutation chromosome	1 0 0 0 1 0 0 1
Child	0 0 1 1 1 1 0 0

Fig: Mutation flipping

A parent is considered and a mutation chromosome is randomly generated. For a 1 in mutation chromosome, the corresponding bit in parent chromosome is flipped (0 to 1 and 1 to 0) and child chromosome is produced. In the case illustrated in Figure above, 1 occurs at 3 places of mutation chromosome, the corresponding bits in parent chromosome are flipped and the child is generated.

2.Gaussian mutation

- **Gaussian Mutation** is a type of mutation typically used in algorithms that work with continuous or real-valued representations. It involves adding a random value drawn from a Gaussian (normal) distribution to an individual's gene values.

  How It Works:

  - For each gene in an individual, a small perturbation is added to the gene's current value.
  - This perturbation is sampled from a Gaussian distribution, which is characterized by its mean (usually zero) and standard deviation (which controls the magnitude of the mutation).

3.Exchange/Swap mutation

- Two positions in the chromosome are randomly selected.
- The values at these two positions are exchanged to create a new individual.

5. Termination

After the reproduction phase, a stopping criterion is applied as a base for termination. The algorithm terminates after the threshold fitness solution is reached. It will identify the final solution as the best solution in the population.

Neural Networks (NN)

A neural network is a processing device, either an algorithm or an actual hardware, whose design was inspired by the design and functioning of animal brains and components thereof. The computing world has a lot to gain from neural networks, also known as artificial neural networks.

The neural networks have the ability to learn by example, which makes them very flexible and powerful. For neural networks, there is no need to devise an algorithm to perform a specific task, that is, there is no need to understand the internal mechanisms of that task. These networks are also well suited for real- time systems because of their fast response and computational times which are because of their parallel architecture.

Before discussing artificial neural networks, let us understand how the human brain works. The human brain is an amazing processor. Its exact workings are still a mystery. The most basic element of the human brain is a specific type of cell, known as neuron, which doesn't regenerate. Because neurons aren't slowly replaced, it is assumed that they provide us with our abilities to remember, think and apply previous experiences to our every action. The human brain comprises about 100 billion neurons. Each typical neuron can connect with up to 200,000 other neurons.

The power of the human mind comes from the sheer numbers of neurons and their multiple interconnections. It also comes from genetic programming and learning. There are over 100 different classes of neurons. The individual neurons are complicated. They have a myriad of parts, subsystems and control mechanisms. They convey information via a host of electrochemical pathways. Together these neurons and their connections form a process which is not binary, not stable, and not synchronous. In short, it is nothing like the currently available electronic computers, or even artificial neural networks.

Biological Neural Network

It is well-known that the human brain consists of a huge number of neurons, approximately 10^{11} with numerous interconnections. A schematic diagram of a biological neuron is shown in Figure below.

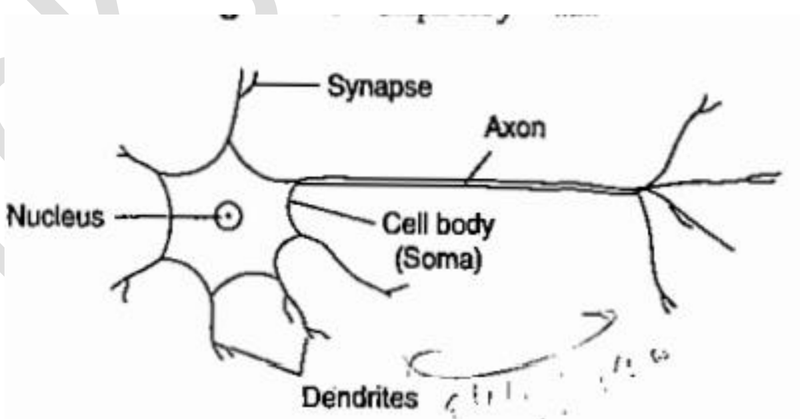


Figure:- Schematic diagram of a biological neuron.

The biological neuron depicted in Figure above consists of three main parts:

- **Soma or Cell Body:** The soma is the central part of the neuron containing the nucleus. It is responsible for maintaining the cell's health and plays a crucial role in processing information received from other neurons. The soma integrates signals from the dendrites and decides whether or not to send a signal down the axon.
- **Dendrites:** Dendrites are tree-like extensions from the soma that receive signals from other neurons. They play a critical role in receiving and processing incoming information. Each neuron can have many dendrites, which increases the surface area for receiving signals.
- **Axon:** The axon is a long, slender projection that conducts electrical impulses away from the soma towards other neurons or muscles. The axon can be very long and is typically covered with a myelin sheath, which helps increase the speed at which electrical signals travel. The end of the axon splits into fine strands. It is found that each strand terminates into a small bulb-like organs called synapse. It is through synapse that the neuron introduces its signals to other nearby neurons. The receiving ends of these synapses on the nearby neurons can be found both on the dendrites and on the cell body. These are approximately 10^4 synapses per neuron in the human brain.

Electric impulses are passed between the synapse and dendrites. This type of signal transmissions involves a chemical process in which specific transmitter substances are released from the sending side of the junction. This results in increase or decrease in the electric potential inside the body of the receiving cell. If the electric potential reaches a threshold, then the receiving cell fires and a pulse or action potential of fixed strength and duration is sent out through the axon to the synaptic junctions of the other cells. After firing, a cell has to wait for a period of time called the refractory period before it can free again. These synapses are said to be inhibitory if they let passing impulses hinder the firing of the receiving cell or excitatory if they let passing impulses cause the firing of the receiving call.

Artificial Neural Networks (ANN)

Neural networks, also known as artificial neural network (ANN) may be defined as an information-processing model that is inspired by the way biological nervous systems, such as the brain, process information. This model tries to replicate only the most basic functions of brain. The key element of ANN is the novel structure of its information processing system. An ANN is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems.

Artificial neural networks, like people, learn by example. Artificial neural network -perform various tasks Such as pattern-matching and classification, optimization function, approximation, vector quantization, and data clustering through a learning process. In biological systems, learning involves adjustments to the synaptic connections that exist between the neurons.

Artificial neural network are those information processing systems, which are constructed and implemented to model the human brain. The main objective of the neural network research is to develop a computational device for modelling the brain to perform various computational tasks at a faster rate than the traditional systems.

ANNs possess large number of highly interconnected processing elements called **nodes or units or neurons,** which usually operate in parallel and are configured in regular architectures.

Each neuron is connected with the other by a connection link. Each connection link is associated with weights which contain information about the input signal. This information is used by the neuron net to solve a particular problem.

ANNs collective behaviour is characterized by their ability to learn, recall and generalize training patterns or data similar to that of a human brain. They have the capability to model networks of original neurons as found in the brain. Thus, the ANN processing elements are called neurons or artificial neurons.

It should be noted that each neuron has an internal state of its own. This internal state is called the **activation or activity level** of neuron, which is the function of the inputs the neuron receives. The activation signal of a neuron is transmitted to other neurons. A neuron can send only one Signal at a time, which can be transmitted to several other neurons.

To depict the basic operation of a neural net, consider a set of neurons, say X1 and X2, transmitting signals to another neuron, Y. Here X1 and X2 are input neurons, which transmit signals, and Y is the output neuron, which receives signals as shown in figure below.

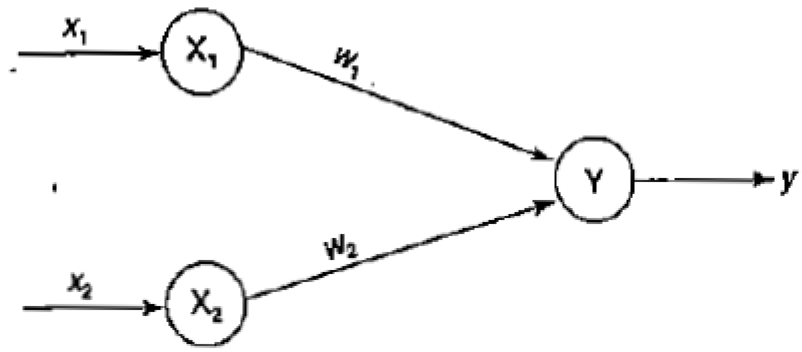


Figure :-Architecture of a simple artificial neuron net.

Here input neurons X1 and X2 are connected to the output neuron Y, over a weighted interconnection links W1 and W2.

For the above simple neuron net architecture, the net input has to be calculated in the following way:

$$y_{in} = x1\ w1 + x2\ w2$$

where x1 and x2 are the activations of the input neurons X1 and X2, i.e. the output of input signals. The output y of the output neuron Y can be obtained by applying activations over the net input, i.e., the function of the net input:

$$y = f(y_{in})$$

Output = Function (net input calculated)

- y_{in}: This represents the net input to the neuron, which is typically the weighted sum of inputs from other neurons or external sources.

- y: The output of the neuron after applying the activation function.

Hence, the function to be applied over the net input is called activation function.

The above calculation of the net input is similar to the calculation of output of a pure linear straight-line equation (y = mx). The neural net of a pure linear equation is as shown in Figure below.

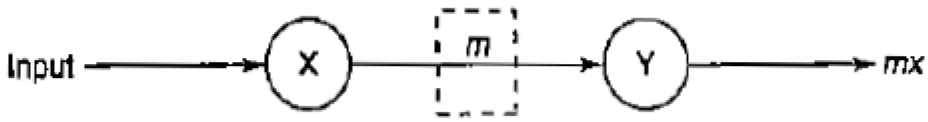


Figure: Neural net of pure linear equation.

Here, to obtain the output y, the slope m is directly multiplied with the input signal. This is a linear equation. Thus, when slope and input are linearly varied, the output is also linearly varied, as shown in Figure below. This shows that the weight involved in the ANN is equivalent to the slope of the linear straight line.

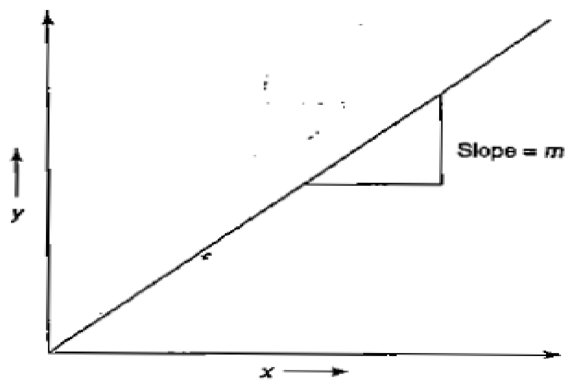


Figure: - Graph for y = mx.

<u>Mathematical Model of ANN</u>

Figure below shows a mathematical representation of the above discussed chemical processing taking place in an artificial neuron.

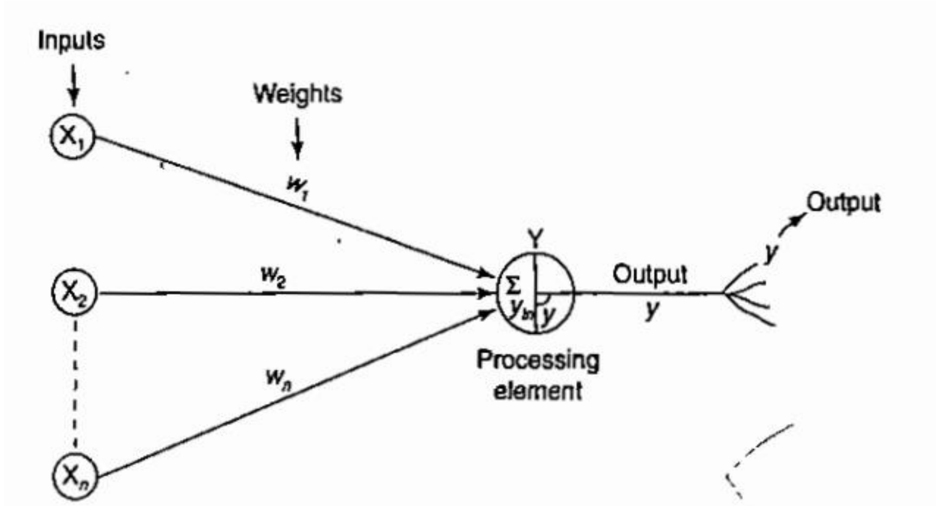


Figure: Mathematical model of artificial neuron.

In this model, the net input is elucidated as

$$y_{in} = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n = \sum_{i=1}^{n} x_i w_i$$

Where i represents the i^th processing element. The activation function is applied over it to calculate the output. The weigh represents the strength of synapse connecting the input and the output neurons. A positive weight corresponds to an excitatory synapse, and a negative weight corresponds to an inhibitory synapse.

The terms associated with the biological neuron and their counterparts in artificial neuron are presented in Table below.

Table: - Terminology relationship between biological and artificial neurons

Biological neuron	Artificial neuron
Cell	Neuron
Dendrites	Weights or interconnections
Soma	Net input
Axon	Output

Characteristics of ANN:

- It is a neutrally implemented mathematical model
- There exists large number of highly interconnected processing elements called neurons in ANN.
- The Interconnections with weighted linkage hold informative knowledge.
- The Input signals arrive at processing elements through connections and connecting weights.
- The Processing elements of ANN have the ability to learn, recall and generalize from the given data by suitable assignment and adjustment of weights.
- The Computational power can be determined by the collective behaviour of neurons and it should be noted that no single neuron carries specific information.

Biological Neural Networks Vs. Artificial Neural Networks (ANN)
(Brain V/S Computer)

A comparison could be made between biological and artificial neurons on the basis of the following criteria:

Speed: -

The cycle time of execution in the ANN is of few nanoseconds whereas in the case of biological neuron is of a few milliseconds. Hence, the artificial neuron modelled using a computer is faster.

Processing: -

Basically, the biological neuron can perform massive parallel operations simultaneously. The artificial neuron can also perform several parallel operations simultaneously, but, in general, the artificial neuron network process is faster than that of the brain.

Size and complexity: -

The total number of neurons in the brain is about 10^{11} and the total number of interconnections is about 10^{15}. Hence, it can be noted that the complexity of the brain is comparatively higher, i.e., the computational work takes places not only in the brain cell body, but also in axon, synapse, ere. On the other hand, the size and complexity of an ANN is based on the chosen application and the network designer. The size and complexity of a biological neuron is more than that of an artificial neuron.

Storage capacity (memory): -

The biological neuron stores the information in its interconnections or in synapse strength but in an artificial neuron it is stored in its contiguous memory locations. In an artificial neuron, the continuous loading of new information may sometimes overload the memory locations. As a result, some of the addresses containing older memory locations may be destroyed.

But in case of the brain, new information can be added in the interconnections by adjusting the strength without destroying the older information. A disadvantage related to brain is that sometimes its memory may fail to recollect the stored information whereas in an artificial neuron, once the information is stored in its locations, it can be retrieved. Owing to these facts, the adaptability is more toward an artificial neuron.

Tolerance: -

The biological neuron assesses fault tolerant capability - whereas the artificial neuron has no fault tolerance. The distributed nature of the biological neurons enables to store and retrieve information even when the interconnections in them get disconnected. Thus, biological neurons are fault tolerant. But in case of artificial neurons, the interconnections gets corrupted if the network interconnections are disconnected. Biological neurons can accept redundancies, which is not possible in artificial neurons. Even when some cells die, the human nervous system appears to be performing with the same efficiency.

Control mechanism: -

In an artificial neuron modelled using a computer, there is a control unit present in Central Processing Unit, which can transfer and control precise scalar values from unit to unit, but there is no such control unit for monitoring in the brain. The strength of a neuron in the brain depends on the active chemicals present and whether neuron connections are strong or weak as a result of structure layer rather than individual synapses. However, the ANN possesses simpler interconnections and is free from chemical actions similar to those raking place in brain (biological neuron). Thus, the control mechanism of an artificial neuron is very simple compared to that of a biological neuron.

BNN Vs. ANN

Term	Brain	Computer
Speed	Execution time is few milliseconds	Execution time is few nano seconds
Processing	Perform massive parallel operations simultaneously	Perform several parallel operations simultaneously. It is faster then biological neuron
Size and complexity	Number of Neuron is 10^{11} and number of interconnections is 10^{15}. So, complexity of brain is higher than computer	It depends on the chosen application and network designer.
Storage capacity	• Information is stored in interconnections or in synapse strength. • New information is stored without destroying old one. • Sometimes fails to recollect information	• Stored in continuous memory location. • Overloading may destroy older locations. • Can be easily retrieved

Tolerance	• Fault tolerant • Store and retrieve information even interconnections fail • Accept redundancies	• No fault tolerance • Information corrupted if the network connections disconnected. • No redundancies
Control mechanism	Depends on active chemicals and neuron connections are strong or weak	CPU Control mechanism is very simple

Types of ANN:

1. **Single-layer feed-forward network**

A network is said to be a feedforward network if no neuron in the output layer is an input to a node in the same layer or in the preceding layer.

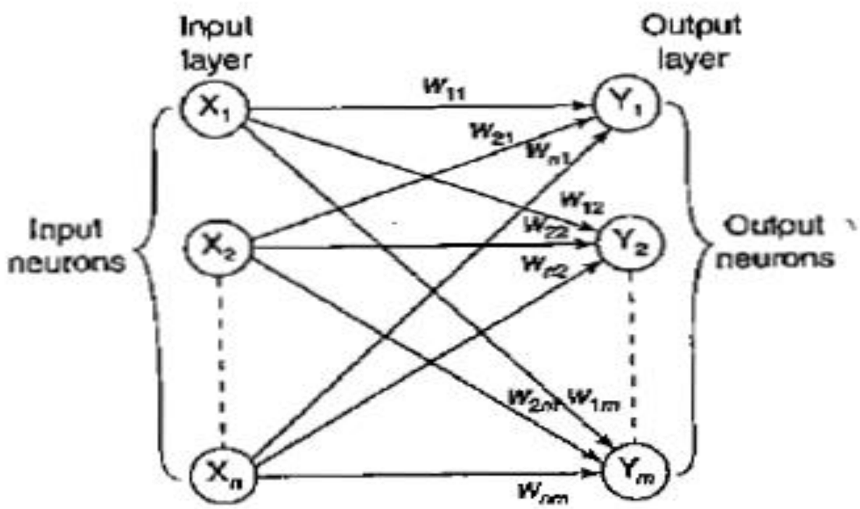


Figure:(A) Single layer feed-forward network.

Basically, neural nets are classified into single-layer or multilayer neural nets. A layer is formed by taking a processing element and combining it with other processing elements. Practically, a layer implies a stage, going stage by stage, i.e., the input stage and the output stage are linked with each other. These linked interconnections lead to the formation of various network architectures.

When a layer of the processing nodes is formed, the inputs can be connected to these nodes with various weights, resulting in a series of outputs. Thus, a single-layer feed-forward network is formed which is shown in above figure.

2. **Multilayer feed-forward network**

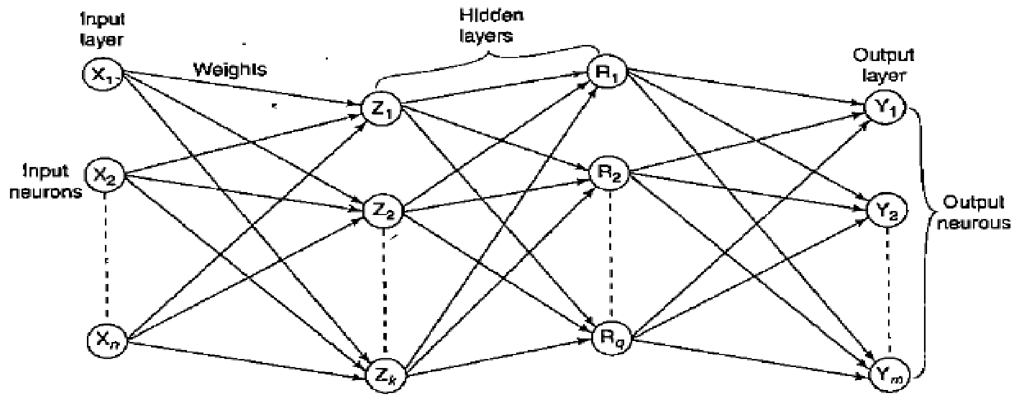


Figure: (B) Multilayer feed-forward network.

A multilayer feed-forward network is formed by the interconnection of several layers. The input layer is that which receives the input and this layer has no function except buffering the input signal. The output layer generates the output of the network. Any layer that is formed between input and output layers is called hidden layer. This hidden layer is internal to the network and has no direct contact with the external environment. It should be noted that there may be zero to several hidden layers in an ANN. More the number of the hidden layers, more is the complexity of the network.

3. **<u>Single node with its own feedback</u>**

when an output can be directed back as inputs to same or preceding layer nodes then it results in the formation of feed-back network which is shown in figure below.

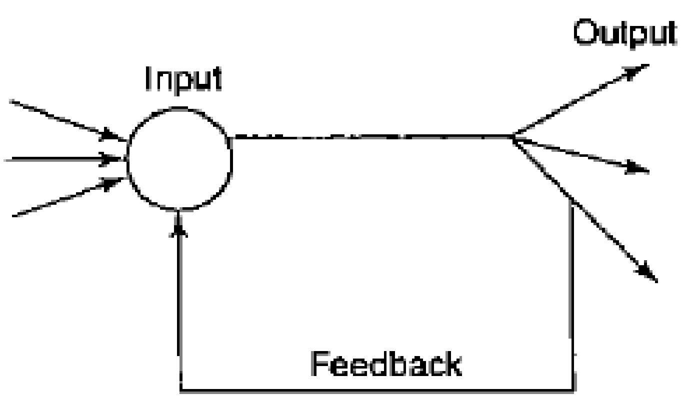


Figure: (C) Single node with own feedback.

4. Single-layer recurrent network

Recurrent networks are feedback networks with closed loop. Figure (C) above shows a simple recurrent neural network having a single neuron with feedback to itself.

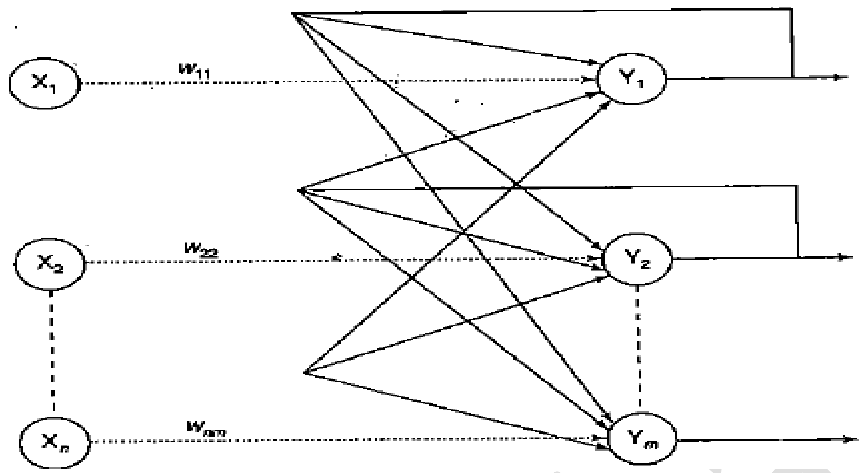


Figure (D) Single-layer recurrent network.

Figure (D) above shows a single layer network with a feedback connection in which a processing element's output can be directed back to the processing element itself or to the other processing element or to both.

5. Multilayer recurrent network.

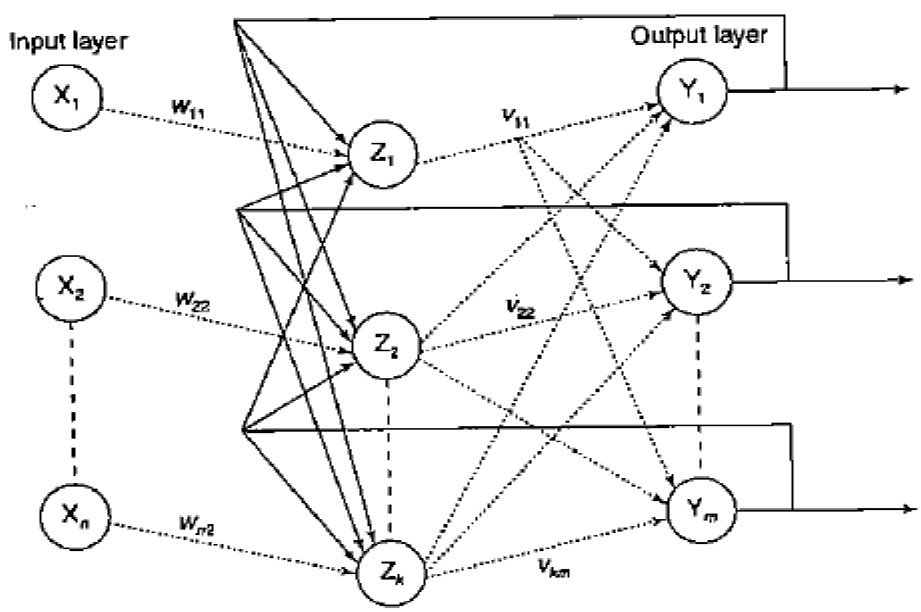


Figure: (E) Multilayer recurrent network.

From Figure (E) above, it can be noted that a processing element output can be directed back to the nodes in a preceding layer, forming a multilayer recurrent network. Also, in these networks, a processing element output can be directed back to the processing element itself and to other processing elements in the same layer.

Application of Artificial Neural Networks

The neural networks have good scope of being used in the following areas:

1. Air traffic control could be automated with the location, altitude, direction and speed of each radar blip taken as input to the network. The output would be the air traffic controller's instruction in response to each blip.
2. Animal behaviour, predator/prey relationships and population cycles may be suitable for analysis by neural networks.
3. Appraisal and valuation of property, buildings, automobiles, machinery, etc. should be an easy task for a neural network.
4. Betting on horse races, stock markets, sporting events, etc. could be based on neural network predictions.
5. Criminal sentencing could be predicted using a large sample of crime details as input and the resulting sentences as output.
6. Complex physical and chemical processes that may involve the interaction of numerous (possibly unknown) mathematical formulas could be modelled heuristically using a neural network.
7. Data mining, cleaning and validation could be achieved by determining which records suspiciously diverge from the pattern of their peers.
8. Direct mail advertisers could use neural network analysis of their databases to decide which customers should be targeted, and avoid wasting money on unlikely targets.
9. Echo patterns from sonar, radar, seismic and magnetic instruments could be used to predict their targets.
10. Econometric modelling based on neural networks should be more realistic than older models based on classical statistics.
11. Employee hiring could be optimized if the neural networks were able to predict which job applicant would show the best job performance.
12. Expert consultants could package their intuitive expertise into a neural network to automate their services.
13. Fraud detection regarding credit cards, insurance or taxes could be automated using a neural network analysis of past incidents.
14. Hand-writing and typewriting could be recognized by imposing a grid over the writing, then each square of the grid becomes an input to the neural network. This is called "Optical Character Recognition."
15. Lake water levels could be predicted based upon precipitation patterns and river/dam flows.
16. Machinery control could be automated by capturing the actions of experienced machine operators into a neural network.
17. Medical diagnosis is an ideal application for neural networks.
18. Medical research relies heavily on classical statistics to analyse research data. Perhaps a neural network should be included in the researcher's tool kit.
19. Music composition has been tried using neural networks. The network is trained to recognize patterns in the pitch and tempo of certain music, and then the network writes its own music.
20. Photos and fingerprints could be recognized by imposing a fine grid over the photo. Each square of the grid becomes an input to the neural network.
21. Recipes and chemical formulations could be optimized based on the predicted outcome of a formula change.
22. Retail inventories could be optimized by predicting demand based on past patterns.

23. River water levels could be predicted based on upstream reports, and time and location of each report.
24. Scheduling of buses, airplanes and elevators could be optimized by predicting demand
25. Staff scheduling requirements for restaurants, retail stores, police stations, banks, etc., could be predicted based on the customer flow, day of week, paydays, holidays, weather, season, etc. stimuli. For example, a football coach must decide whether to kick, pass or run on the last down. The inputs for this decision include score, time, field location, yards to first down, etc.
26. Strategies for games, business and war can be captured by analysing the expert players response to given

Learning by Training ANN

The main property of an ANN is its capability to learn. Learning or training is a process by means of which a neural network adapts itself to a stimulus by making proper parameter adjustment, resulting in the production of desired response. Broadly, there are two kinds learning in ANNs:

1. **Parameter learning:** it updates the connecting weights in a neural net.
2. **Structured learning:** It focuses on the change in network structure (which includes the number of processing elements as well as their connection types).

The above two types of learning can be performed simultaneously or separately. Apart from these two categories of learning, the learning in an ANN can be generally classified into three categories as:

1. Supervised learning;
2. Unsupervised learning
3. Reinforcement learning.

Let us discuss these learning types in detail.

Supervised learning;

The supervised learning is performed with the help of a teacher. Let us take the example of the learning process of a small child. The child doesn't know how to read/write. He/she is being taught by the parents at home and by the teacher in school. The children are trained and modelled to recognize the alphabets, numerals, etc. Theirs’s each and every action is supervised by a teacher. Actually, a child works on the basis of the output that he/she has to produce. All these real-time events involve supervised learning methodology.

Similarly, in ANNs following the supervised learning, each input vector requires a corresponding target vector, which represents the desired output. The input vector along with the target vector is called training pair. The network here is informed precisely about what should be emitted as output. The block diagram below depicts the working of supervised learning network.

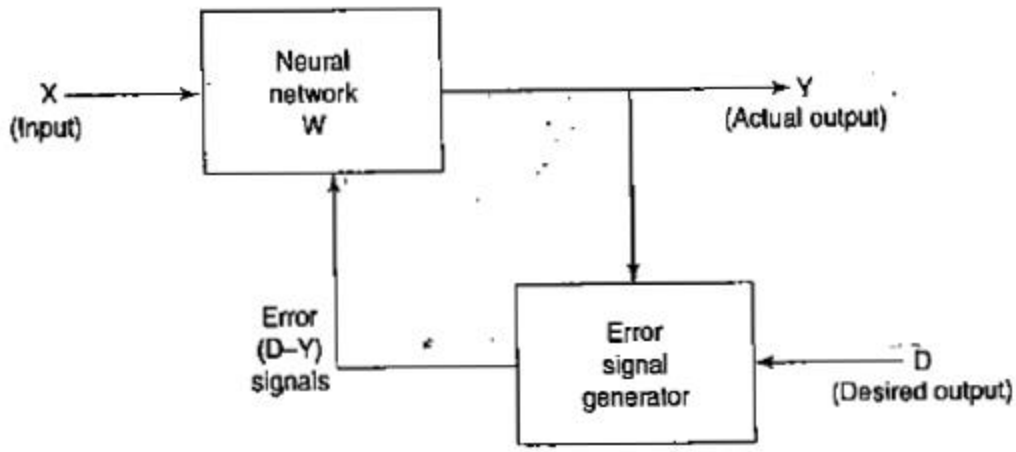


Figure: Block diagram of Supervised learning.

During training. the input vector is presented to the network, which results in an output vector. This output vector is the actual output vector. Then the actual output vector is compared with the desired (target) output ·vector. If there exists a difference between the two output vectors then an error signal is generated by the network.

This error signal is used for adjustment of weights until the actual output matches the desired (target) output.

In this type of training, a supervisor or teacher is required for error minimization. Hence, the network trained by this method is said to be using supervised training methodology. In supervised learning it is assumed that the correct "target" output values are known for each input pattern.

Unsupervised learning

The learning here is performed **without** the help of a teacher. Consider the learning process of a tadpole, it learns by itself, that is, a child fish learns to swim by itself, it is not taught by its mother. Thus, its learning process is independent and is not supervised by a teacher.

The block 1agram of unsupervised learning is shown in Figure below

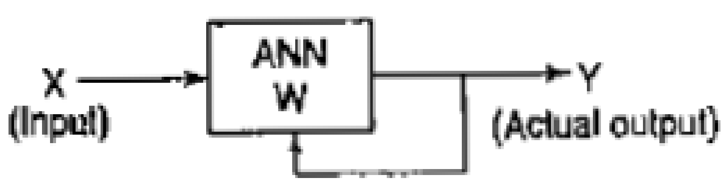


Figure: Unsupervised learning.

In unsupervised learning, the input vectors of similar type are grouped without the use of training data to specify how a member of each group looks or to which group a number belongs. In the training process, the network receives the input patterns and organizes these patterns to form clusters.

When a new input pattern is applied, the neural network gives an output response indicating the class to which the input pattern belongs. If for an input, a pattern class cannot be found then a new class is generated.

From Figure above the network itself discover patterns, regularities, features or categories from the input data and relations for the input data over output. While discovering all these features, the network undergoes change in Its parameters. This process is called **“self-Organizing”** in which exact clusters will be formed by discovering similarities and dissimilarities among the objects.

Reinforcement learning

This learning process is similar to supervised learning. In the case of supervised learning, the correct target output values are known for each input pattern. But, in some cases, less information might be available.

For example, the network might be told that its actual output is only "50% correct" or so. Thus, here only critic information is available, not the exact information. **The learning based on this critic information is called reinforcement learning and the feedback sent is called reinforcement signal.** The block diagram of reinforcement learning is shown in Figure below.

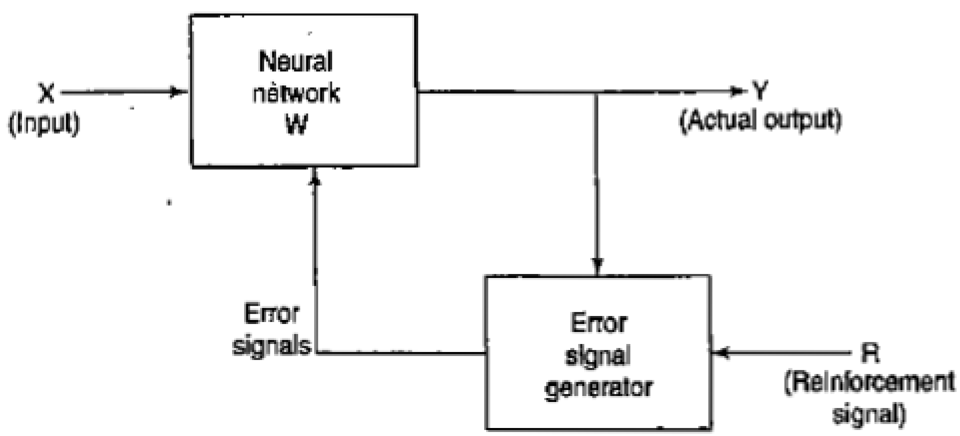


Figure: Reinforcement learning.

The reinforcement learning is a form of supervised learning because the network receives some feedback from its environment. However, the feedback obtained here is only evaluative and not instructive. The external reinforcement signals are processed in the critic signal generator, and obtained critic signals are sent to the ANN for adjustment of weights properly so as to get better critic feedback in future.

The reinforcement learning is also called learning with a critic as opposed to learning with a teacher, which indicates supervised learning.

Hebbian Learning

Donald Hebb stated in 1949 that in the brain, the learning is performed by the change in the synaptic gap(strength). Hebb explained it: "When an axon of cell A is near enough to excite cell B, and repeatedly or permanently takes place in firing it, some growth process or metabolic change take place in one or both the cells such that A's efficiency, as one of the cell's firing B, is increased."

That is, In Biological Basis when an axon of one neuron repeatedly activates another neuron, chemical and structural changes occur in the synapse that increase the efficiency of this connection. These changes can involve increases in neurotransmitter release, changes in receptor density on the postsynaptic neuron, or even the growth of new synaptic connections.

According to the Hebb rule, the weight vector is found to increase proportionately to the product of the input and the learning signal. Here the learning signal is equal to the neuron's output. In Hebb learning, if two interconnected neurons are 'on' simultaneously then the weights associated with these neurons can be increased by the modification made in their synaptic gap (strength).

Hence, Hebb suggested that learning occurs when the repeated and persistent activation of one neuron by another leads to an increase in the synaptic strength between them. This is often viewed as a mechanism for associative learning, where associations between inputs (like sensory experiences) are formed and stored.

In artificial neural networks, Hebbian learning can be mathematically expressed as:

$$\Delta w_{ij} = \eta \cdot x_i \cdot y_j$$

- Δwij is the change in the weight between neuron i and neuron j.
- η(eta) is the learning rate.
- xi is the input from neuron i.
- yj is the output of neuron j.

The updated weight can be expressed as:

$$W(new)=W(old)+\eta.x_i.yj$$

Where:

- Wnew: is the updated weight after the learning step.
- Wold: is the previous weight before the update.
- η is the learning rate, which controls the magnitude of the weight update.
- xi is the input from neuron i (pre-synaptic neuron).
- yi, is the output of the post-synaptic neuron j, which serves as the learning signal in Hebbian learning.

The flowchart for the training algorithm of Hebb network is given in Figure below.

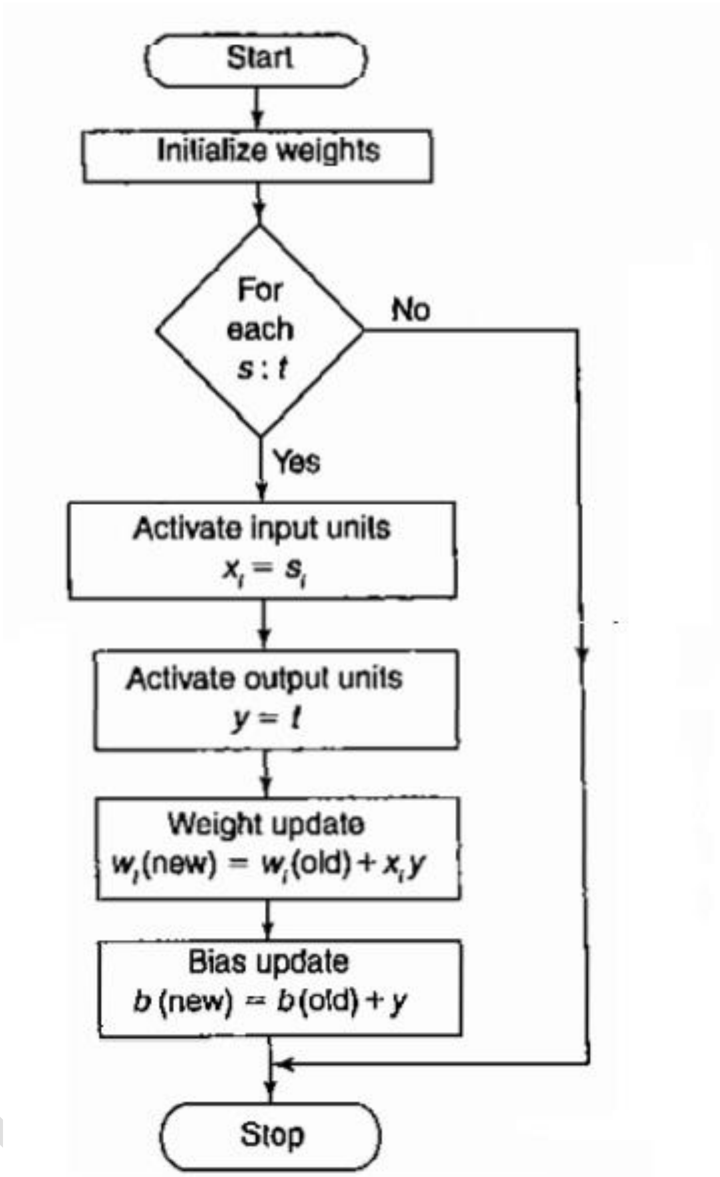


Figure: Flowchart of Hebb training algorithm.

In Figure above, s: t refers to each training input and target output pair. Till there exists a pair of training input and target output, the training process takes place; else, it is stopped.

The training algorithm of Hebb network is given below:

Step 0:

First initialize the weights. Basically, in this network they may be zero,
i.e., Wi = 0 for i = 1 to n where "n" may be the total number of input neurons.

Step 1:

Steps 2-4 have to performed for each input training vector and target output pair, s: t.

Step 2:

Input unit's activations are set. Generally, the activation function of input layer is identify function:

$x_i = s_i$ for i = 1 to n

Step 3: Output unit's activations are set:

y = t

Step 4: Weight adjustments and bias adjudgments are performed:

$$w_i(\text{new}) = w_i(\text{old}) + x_i y$$
$$b(\text{new}) = b(\text{old}) + y$$

The above five steps complete the algorithmic process. In step 4, the weight updation formula can also be given in vector form as

$$w(\text{new}) = w(\text{old}) + xy$$

Here the change in weight can be expressed as·

$$\Delta w = xy$$

As a result,

$$w(\text{new}) = w(\text{old}) + \Delta w$$

The Hebb rule can be used for pattern association, pattern categorization, pattern classification and over a range of other areas.

Perceptron Learning

1. Perceptron networks come under single-layer feed-forward networks and are also called simple perceptron.
2. The perceptron network consists of three units, namely, **sensory unit (input unit), associator unit (hidden unit), and response unit (output unit).**
3. The sensory units are connected to associator units with fixed weights having values 1, 0 or -l, which are assigned at random.
4. The binary activation function is used in sensory unit and associator unit.
5. The response unit has an activation of l, 0 or -1. The binary step with fixed threshold θ is used as activation for associator. The output signals that are send from the associator unit to the response unit are only binary.
6. The output of the perceptron network is given by

   y = f(yin)

   where f(yin)is activation function and is defined as

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \le y_{in} \le \theta \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

7. The perceptron learning rule is used in the weight updation between the associator unit and the response unit. For each training input, the net will calculate the response and it will determine whether or not an error has occurred.
8. The error calculation is based on the comparison of the values with of the targets with those of the calculated outputs.
9. The weights on the connections from the units that send the nonzero signal will get adjusted suitably.
10. The weights will be adjusted on the basis of the learning rule if an error has occurred for a particular training pattern.

$$\mathbf{W_i(new)=w_i(old)+\alpha\ t\ x_i}$$
$$\mathbf{b(new)=b(old)+\ \alpha\ t}$$

where t is the target value and **α** is the learning rate.
If no error occurs, there is no weight updating and hence the training process may be stopped.

A perceptron network with its three units is shown in Figure below.

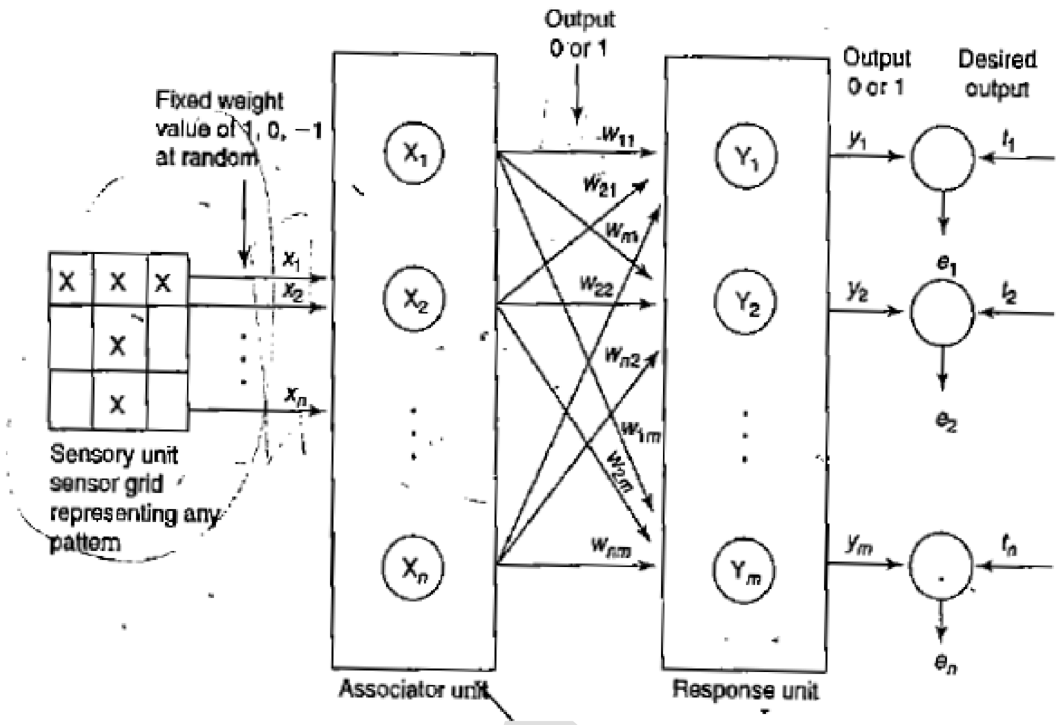


Figure: - Original perceptron network.

- As shown in Figure above. a sensory unit can be a two-dimensional matrix of 400 photodetectors. These detectors provide a binary (0) electrical signal if the input signal is found to exceed the certain value of threshold. Also, these detectors are connected randomly with the associator unit.

- The associator unit is found to consist of a set of sub circuits called feature predicates. The feature predicates are hard-wired to detect the specific feature of a pattern and are equivalent to the feature detectors.

- For a particular feature, each predicate is examined with a few or all of the responses of the sensory unit. It can be found that the results from the predicate units are also binary (0 1).
- The last unit, i.e. response unit, contains the Pattern-recognizers or perceptron the weights present the input layers are all fixed, while the weights on the response unit are trainable.

- In the original perceptron network, the output obtained from the associator unit is a binary vector, and hence that output can be taken as input signal to the response unit and classification can be performed. Here only the weights between the associator unit and the output unit can be adjusted, and the weight between the sensory and associator units are fixed. As a result, the discussion of the network is limited to a single portion. Thus, the associator unit behaves like the input unit.

Perceptron Learning Rule

In case of the perceptron learning rule, the learning signal is the difference between the desired and actual response of a neuron.

Consider a finite "n" number of input training vectors, with their associated target(desired) values x(n) and t(n), where "n" ranges from 1 to N. The target is either + 1 or -1. The output "y" is obtained on the basis of the net input calculated and activation function being applied over the net input.

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \leq y_{in} \leq \theta \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

The weight updation in case of perceptron learning is…..

If y≠ t , then

W(new) = w(old) + $\alpha\, t\, x$ **{ α - learning rate)**

else, we have

w(new) = w(old)

Architecture

A simple perceptron network architecture is shown in Figure below.

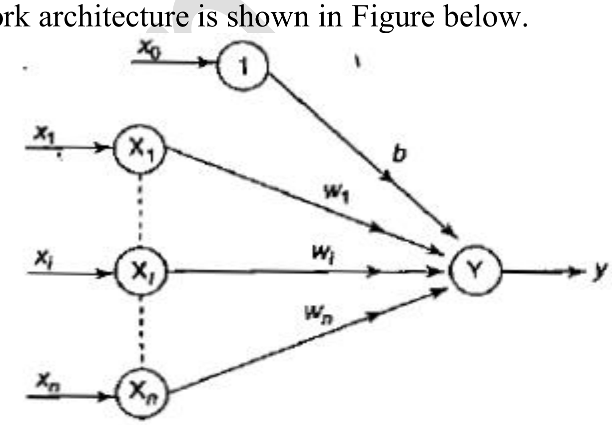


Figure : - Single classification perceptron network.

In Figure above, there are n input neurons, 1 output neuron and a bias. The input-layer and output layer neurons are connected through a directed communication link, which is associated with weights. The goal of the perceptron net is to classify the input pattern as a member or not a member to a particular class.

Training Algorithm

The algorithm of a perceptron network is as follows:

Step 0: Initialize all weights and the bis (for easy calculation they can be set to zero). Also initialize the learning rate α (0< α<= 1). For simplicity α is set to 1.

Step 1: Perform Steps 2-6 until the final stopping condition is false.

Step 2: Perform Steps 3-5 for each training pair indicated by s:t.

Step 3: The input layer containing input units is applied with identity activation functions:

$$x_i = s_i$$

Step 4: Calculate the output of the network. To do so, first obtain the net input:

$$y_{in} = b + \sum_{i=1}^{n} x_i w_i$$

where "n" is the number of input neurons in the input layer. Then apply activations over the net input calculated to obtain the output:

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \leq y_{in} \leq \theta \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

Step 5: Weight and bias adjustment: Compare the value of the actual (calculated) output and desired (target) output.

If $y \neq t$, then

$$w_i(\text{new}) = w_i(\text{old}) + \alpha t x_i$$

$$b(\text{new}) = b(\text{old}) + \alpha t$$

else, we have

$$w_i(\text{new}) = w_i(\text{old})$$

$$b(\text{new}) = b(\text{old})$$

Step 6: Train the network until there is no weight change. This is the stopping condition for the network. If this condition is not met, then start again from Step 2.

Back Propagation algorithm

- The Backpropagation algorithm is a supervised learning method for multilayer feed-forward networks from the field of Artificial Neural Networks.
- The principle of the backpropagation approach is to model a given function by modifying internal weightings of input signals to produce an expected output signal. The system is trained using a supervised learning method, where the error between the system's output and a known expected output is presented to the system and used to modify its internal state.
- Technically, the backpropagation algorithm is a method for training the weights in a multilayer feed-forward neural network. As such, it requires a network structure to be defined of one or more layers where one layer is fully connected to the next layer. A standard network structure is one input layer, one hidden layer, and one output layer.
- The Backpropagation algorithm looks for the minimum value of the error function in weight space using a technique called the delta rule or gradient descent. The weights that minimize the error function is then considered to be a solution to the learning problem.

The training of the BPN is done in three stages…

- The feed-forward of the input training pattern,
- The calculation and back-propagation of the error, and
- Updation of weights.

Architecture

A back-propagation neural network is a multilayer feed forward neural network consisting of an input layer, a hidden layer and an output layer. The neurons present in the hidden and output layers have biases, which are the connections from the units whose activation is always 1. The bias terms also act as weights.

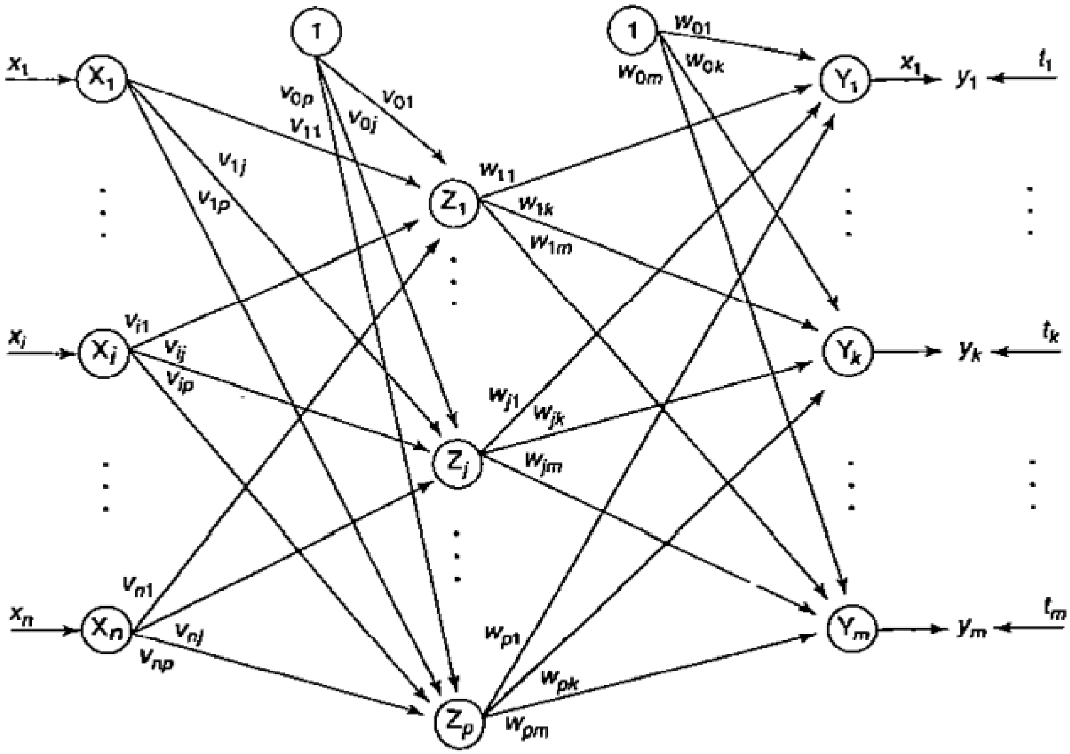


Fig: Architecture of a back-propagation network

Figure above shows the architecture of a BPN, depicting only the direction of information flow for the feed-forward phase. During the back propagation phase of learning, signals are sent in the reverse direction.

The inputs sent to the BPN and the output obtained from the net could be either binary (0,1) or bipolar (-1, + 1). The activation function could be any function which increases monotonically and is also differentiable.

<u>Flowchart for Training Process</u>

The flowchart for the training process using a BPN is shown in Figure below.

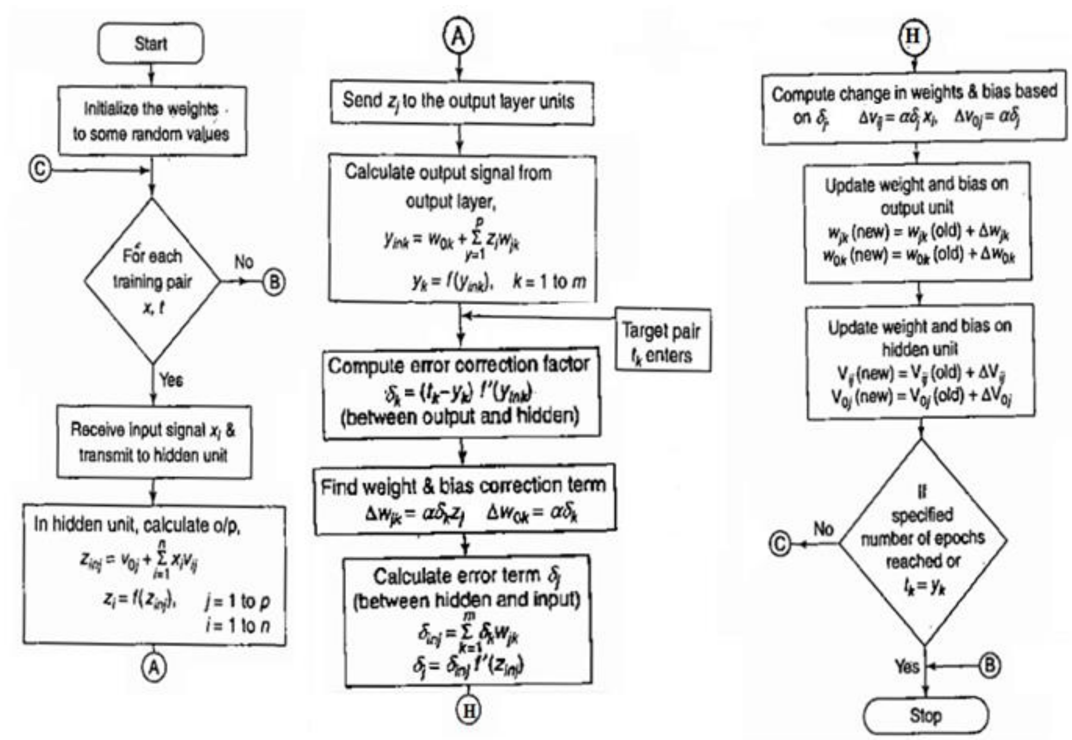


Fig: flow chart for back propagation network training

The terminologies used in the flowchart and in the training, algorithm are as follows:
x = input training vector (x1, ... ,xj, ... , xn)
t = target output vector (t1, ..., tk, ... , tm)
α = learning rate parameter
x_i= input unit i. (Since the input layer uses identity activation function, the input and output signals here are same.)

v_{0j} = bias on j[th] hidden unit

w_{0k} = bias on k[th] output unit

z_j = The hidden unit j.

The net input to z_j is

$$z_{inj} = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

and the output is

$$z_j = f(z_{inj})$$

y_k = output unit k.
The net input to Y_k is

$$y_{ink} = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

and the output is

$$y_k = f(y_{ink})$$

δk = error correction weight adjustment for ω_{jk} that is due to an error at output unit Y_k. which is back-propagated to the hidden units that feed into y_k

δj= error correction weight adjustment for V_{ij} that is due to the back-propagation of error to the hidden unit z_j

Also, it should be noted that the commonly used activation functions are binary sigmoidal and bipolar sigmoidal activation functions. These functions are used in the BPN because of the following characteristics:

I. Continuity
II. Differentiability
III. Nondecreasing monotony

The range of binary sigmoidal is from 0 to 1, and for bipolar sigmoid it is from -1 to+ 1.

<u>Training Algorithm</u>

The error back-propagation learning algorithm can be outlined in the following algorithm:
Step 0: Initialize weights and learning rate (take some small random values).
Step 1: Perform Steps 2-9 when stopping condition is false.
Step 2: Perform Steps 3-8 for each training pair

Feed forward phase (Phase I)
Step 3: Each input unit receives input signal xi and sends it to the hidden unit (i = i to n).
Step 4: Each hidden unit Z_j (j = 1 to p) sums its Weighted input signals to calculate net input:

$$z_{inj} = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

Calculate output of the hidden unit by applying its activation functions over Z_{inj} (binary or bipolar sigmoidal activation function):

$$z_j = f(z_{inj})$$

and send the output signal from the hidden unit to the input of output layer units.

Step 5: For each output y_k (k = 1 to m),_calculate the net input:

$$y_{ink} = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

and apply the activation function to compute output signal –

$$y_k = f(y_{ink})$$

Back-propagation of error (Phase II):
Step 6: Each output unit y_k (k=1 to m) receives a target pattern to the input training pattern and computes the error correction term:

$$\delta_k = (t_k - y_k) f'(y_{ink})$$

On the basis of the calculated error correction term, update the change in weights and bias:

$$\Delta w_{jk} = \alpha \delta_k z_j; \quad \Delta w_{0k} = \alpha \delta_k$$

Also, send δk to the hidden layer backwards.

Step 7: Each hidden unit (Z_j, j = 1 to p) sums its delta inputs from the output units:

$$\delta_{inj} = \sum_{k=1}^{m} \delta_k w_{jk}$$

The term $\boldsymbol{\delta_{inj}}$ gets multiplied with the derivative of $\mathbf{F(Z_{ink})}$ to calculate the error term:

$$\delta_j = \delta_{inj} f'(z_{inj})$$

On the basis of the calculated $\boldsymbol{\delta_j}$ update the change in weights and bias:

$$\Delta v_{ij} = \alpha \delta_j x_i; \quad \Delta v_{0j} = \alpha \delta_j$$

Weight and bias updation (Phase iii):

Step 8: Each output unit ($\mathbf{y_k}$, k = 1 to m) updates the bias and weights:

$$w_{jk}(\text{new}) = w_{jk}(\text{old}) + \Delta w_{jk}$$
$$w_{0k}(\text{new}) = w_{0k}(\text{old}) + \Delta w_{0k}$$

Each hidden unit ($\mathbf{Z_j}$, j = 1 top) updates its bias and weights:

$$v_{ij}(\text{new}) = v_{ij}(\text{old}) + \Delta v_{ij}$$
$$v_{0j}(\text{new}) = v_{0j}(\text{old}) + \Delta v_{0j}$$

Step 9: Check for the stopping condition. The stopping condition may be certain number of epochs reached or when the actual output equals the target output.

The above algorithm uses the incremental approach for updation of weights, i.e., the weights are being changed immediately after a training pattern is presented.

========================= End Unit 5=============================

Unit V: Machine Learning (9 Hrs.)

5.1. Introduction to Machine Learning, Concepts of Learning, Supervised, Unsupervised and Reinforcement Learning

5.2. Statistical-based Learning: Naive Bayes Model

5.3. Learning by Genetic Algorithm

5.4. Learning with Neural Networks: Introduction, Biological Neural Networks Vs. Artificial Neural Networks (ANN), Mathematical Model of ANN, Types of ANN: Feed-forward, Recurrent, Single Layered, Multi-Layered, Application of Artificial Neural Networks, Learning by Training ANN, Supervised vs. Unsupervised Learning, Hebbian Learning, Perceptron Learning, Back-propagation Learning