Introduction: Based on the information about the fn's of the brain, a new technology Artificial Neural N/ws was started. We know our brain can process information quickly and accurately. You can recognise your friend's voice in a noisy railway station. How is the brain able to process the voice signal added with noise and retrieve the original signal?

Amazing isn't it.

→ Can we duplicate this amazing process through a m/c?

→ Can we make a m/c to duplicate some learning habits of human?

→ Can a m/c be made to learn from experience?

So the answers is Neural N/w!

What is Neural Network!- An artificial neural N/w is an information processing system that has been developed as generalization of mathematical model of human Cognition (faculty of knowing).

A neural N/w is a network of interconnected neurons, inspired from the studies of the biological nervous system.

In other words, neural N/w fn's in a way similar to the human brain. The fn of a neural N/w is to produce an output pattern when presented with an I/P pattern.

Historical Perspective:- Humans have always wondered about their own thoughts. This self-referential inquiry, the mind thinking of itself, may be a uniquely human characteristic.

Neurobiologists and neuroanatomists have made substantial progress. Painstakingly mapping out the structure and functions of human brain, they came to understand much of the brain's "wiring" but little of its operation. As their knowledge grew, the complexity was found to be staggering.

The improved understanding of the functioning of the neuron and the pattern of its interconnections has allowed researchers to produce mathematical models to test their theories.

Experiments can now be conducted on digital computers without involving human or animal subjects, thereby solving many practical and ethical problems.

Along with the progress in neuroanatomy and neurophysiology, psychologists were developing models of human learning one such models which has proved most fruitful, was that of D.O. Hebb, who in 1949 proposed a learning law that become the starting point for artificial neural n/w training algorithms. Augmented today by many other methods, it showed scientists of that era how a network of neurons could exhibit learning behaviour.

In 1950s and 1960s, a group of researchers combined these biological and psychological insight to produce the first artificial neural networks

Initially implemented as electronic circuits, they were later converted to the more flexible medium of Computer Simulation. Early successes produced a burst of activity and optimism. Marvin Minsky, Frank Rosenblatt and others developed networks consisting of a single layer of artificial neurons. Often called perceptrons, they were applied to such diverse problems as weather prediction, electrocardiogram analysis and artificial vision. It seemed for a time that the key to intelligence had been found, reproducing the human brain was only a matter of constructing a large enough network.

This illusion was soon dispelled. Networks failed to solve problems superficially similar to those they had been successful in solving. These unexplained failures launched a period of intense analysis. Marvin Minsky, Carefully applying mathematical techniques developed rigorous theorems regarding network operation. His research field led to the publication of book Perceptrons in which he and Seymor proved that the single layer networks then in use were theoretically incapable of solving many simple problems, including the f^n performed by simple exclusive OR gate.

Minsky's brilliance, rigor and prestige gave book great capability: its conclusions were unassailable.

Nevertheless, a few dedicated scientists such as Teuvo Kohonen, James Anderson continued their efforts. Early 1980's is found scattered among wide variety of journals, some of which rather obscure.

Gradually, a theoritical foundation emerged, upon which the more powerful multilayer N/ws of today are being constructed. Minsky's appraisal has proven excessively pessimistic; n/ws are now routinely solving many of the problems that he posed in his book.

In the past few years, Theory has been translated into application and new Corporations dedicated to the Commercialization of the technology have appeared. There has been an explosive increase in the amount of research activity with four major Conventions in 1987 in the field of artificial neural N/ws and over 500 technical papers published, the growth rate has been phenomenal.

Characteristics of Artificial Neural Networks:

1 Artificial neural N/ws are biologically inspired That is, they are composed of elements that perform in a manner that is analogous to most elementary f^ns of the biological neuron. These elements are then organized in a way that may related to anatomy of the brain.

Despite this superficial resemblance, ANNs exhibits a surprising no. of the brain's characteristics. For example, they learn from experience, generalize from previous examples to new ones and abstract essential characteristics from inputs containing irrelevant data.

2) Learning:- ANNs can modify their behaviour in response to their environment. This factor more than any others, is responsible for the interest they have received. Shown a set

of inputs, They self-adjust to produce consistent responses. A wide variety of training algorithms has been developed, each with its own strengths and weaknesses.

3) Generalization:- Once trained, a n/w's response can be, to a degree, insensitive to minor variations in its input. This ability to see through noise and distortion to the pattern that lies within is vital to pattern recognition in a real-world environment. Overcoming the literalmindedness in real world environment of the conventional computer, it produces a system that can deal with the imperfect world in which we live. It is important to note that the ANNs generalizes automatically as a result of its structure, not by using human intelligence embedded in the form of ad hoc computer programs.

4) Abstraction:- Some artificial neural N/Ws are capable of abstracting the essence of a set of inputs. For example, a network can be trained on sequence of distorted versions of the letter A. After adequate training, application of such a distorted example will cause the network to produce a perfectly formed letter. In one sense, it has to learned to produce something that it has never seen before.

This ability to extract an ideal from imperfect inputs raises interesting philosophical issues.

Applicability:- ANN are not a panacea. They are clearly unsuited to such tasks as calculating the payroll. It appears that they

will, however become the preferred technique for a large class of pattern recognition tasks that Conventional Computers do poorly if at all.

Applications of Neural Networks:- There have been many impressive demonstrations of artificial neural networks. A few areas where neural networks are currently applied are mentioned below.

1) Classification:- A neural network can discover the distinguishing features needed to perform a classification task. classification is the assignment of each object to a specific class, which is important aspect in image classification. Neural Networks have been used succesfully in a large no of classification task which includes.

(a) Recognition of printed or handwritten characters
(b) Classification of SONAR and RADAR signals.

2) Signal processing:- In digital Communication Systems, distorted signals often inter-signal interference. One of the first Commercial applications of neural networks was to suppress noise Cancellation and it was implemented by Widrow using ADALINE. The ADALINE is trained to remove the noise from the telephone line signal.

3) Speech Recognition:- In recent years, speech recognition has received enormous attention. It involves three moduls namely, The front end which samples the speech signals and extract the data, The word processor which is used for finding the

probability of words in the vocabulary that match the features of spoken words and the sentence processor which determines if the recognized word makes sense in the sentence. Real time speech recognition system is crucial since it requires a large vocabulary database and large RAM is also required. Kohonen's "Phonetic Typewriter" was an early application in speech recognition, which automatically recognized vocabulary and converted it to text. Neural N/ws remove the noise inherent in the speech.

4). Medicine :- Anderson developed an autoassociative n/w to store large amount of medical records, each of which includes information on the symptoms, diagonisis and treatment of a particular case. When a trained N/w is presented with a set of symptoms, the net finds the best diagonisis and treatment.

5). Intelligent Control :- A problem of concern in industrial motor control is the ability to predict system failure. Motor failure depends on specific motor parameters, transient currents and motor positioning. Neural N/ws can learned to employed the motor-current variants as well as installation characteristics. 80% to 90% prediction rate has been reported. Many of neural N/w applications merge in Robotics. The simplest tasks performed are arm movement, object recognition and object manipulation. Neural networks have been used in many vehicular applications including trains and automatic gear transmission in cars.

6). Function Approximation :- Many Computational models can be described as functions mapping the input vectors to numerical outputs.. Neural N/ws are employed to Construct the f^n that generates approximately the same o/p for given input vector based on available training Data.

7). Financial Forecasting :- The idea that if one dealer can predict market values or assess risk better than his Competitors even by a very small amount, leads to the expectation of huge financial benefit Much of the work in this area is based on the so called "time Series. prediction.

In Contrast with time Series prediction, neural N/w have also been used in financial modeling which aims at testing Theories about the dynamics of Mkt.

8) Condition Monitoring :- The ability of neural nets to receive input from a large no of Sensors and to learn to assess the significance of them is a major advantage in Condition monitoring. The Sensitivity to novelty is almost an emergent Property of most neural Systems. This has obvious applications in the detection of departures from normality. Time dependent behaviour of nets Can be also be used in predicting the occurance of malfunction ahead of time. Application under this range from health monitoring of turbo m/c to gear boxes, jets and internal Combustion engines.

9). Process Monitoring and Control :- Neurocontrol principles can be applied to various areas of the process control industry. Neural N/w techniques come into their own when the link b/w the measurement and operation parameter selection can't be obtained by analysis. Then technique can be applied in processes such as chemical, biochemical, pharmaceutical, water purification, food and beverage production, power distribution and oil and gas industries. (5)

10) Neuro Forecasting :- A major of neural nets is the prediction of prospective long time foreign exchange rates. Neural N/ws provide greater flexibility in this regard compared to the traditional methods.

11). Pattern Analysis :- This application works in various areas such as online industrial plant monitoring, smart sensor systems and industrial image analysis as may be required in inspection and control tasks. Other classical problems of pattern analysis such as real time audio bandwidth systems, images and data management can also be tackled with the help of neural N/ws.

→ ANN in Contrast have usually been limited to 10,000 units with hundreds of Connections per unit.

→ long term memory resides in the Synapses or weights and short-term memory Corresponds to signals sent by the neurons.

→ The human brain is extremely large for a neural N/W Containing 10^{11} neurons with as many as 10^{4} interconnections each.

→ This System leads to more possible neural assemblies and activity pattern That Could in principle be used in neural function.

3). Fault Tolerance :-

ANN

→ Learning takes place by a formulated set of rules and hence the system is less faulty. The Processing elements receive many signals and sum up the weighted inputs. The networks can be retrained in case of significant damage.

→ Redundancy can increase the reliability of the System, allowing it to fn even when Some of the neural units are destroyed.

→ Generalization of pattern is more difficult. Back propagation organizes feature detectors in middle layers of System with Three or more layers.

Biological Neuron.

→ Learning takes place arbitrarily and hence the System is bound for failure. Biological n/ws. are fault tolerance and even in a traumatic loss, other neurons can take over the function of damaged Cells.

→ Redundancy is used when Some Cells die, the nervous System appears to perform the Same. It Can Counteract Sources of noise in biological Systems.

→ Generalisation becomes easier in biological n/ws. This System extracts an abstract of special features from Sensory inputs Corresponding to feature detectors.

4) Control Mechanism

ANN

→ The strength of the neuron depends on whether it is active or inactive and have relatively simpler interconn's.

→ ANN can transfer precise scalar values from unit to unit.

→ Artificial n/w have fully interconnected stabs are often layered.

Biological N/ws.

→ The strength of the neuron depends on active chemicals present and connections are stronger or weaker as a result of structure layer rather than individual synapses.

→ Biological impulse transmission takes place at any time and its timing is determined by incoming signals.

→ Biological systems generally have predetermined wiring at the system level.

Fundamentals of Neural N/ws:-

The biological prototype:- Neural N/w architectures are motivated by models of human brain and nerve cells. Our current knowledge of human brain is limited to its anatomical and physiological information. Neuron (from Greek, meaning nerve cell) is the fundamental unit of the brain. The neuron is a complex biochemical and electrical signal processing unit that receives and combines signals from many other neurons through filamentary input paths i the dendrites. A neuron in the human brain is shown in figure 1.1

A biological neuron has three types of components namely:- Dendrites, Soma and axon, Synapse.

⇒ **Dendrites:-** are bunched into highly complex "dendritic tree", which have an enormous total surface area. The dendrites receives signals from other neurons

⇒ **Soma:-** The dendrites trees are connected with the main body of neuron called Soma The Soma has pyramidal or Cylindrical shape. The Soma sums the incoming signals. When sufficient input is received, the cells fires.

⇒ **Axon:-** The O/P area of the neuron is a long fibre called axon. The impulse signal triggered by the cell is transmitted over the axon to other cells.

⇒ Synapses: The Connecting point b/w a neuron's axon and another neuron's dendrite is called Synapse. (Greek: Contact). The impulse signals are then transmitted across a Synaptic gap by means of a Chemical process.

→ A single neuron may have 1000 to 10000 Synapses and may be Connected with around 1000 neurons.

→ There are 100 billion neurons in our brain, and each neuron has 1000 dendrites.

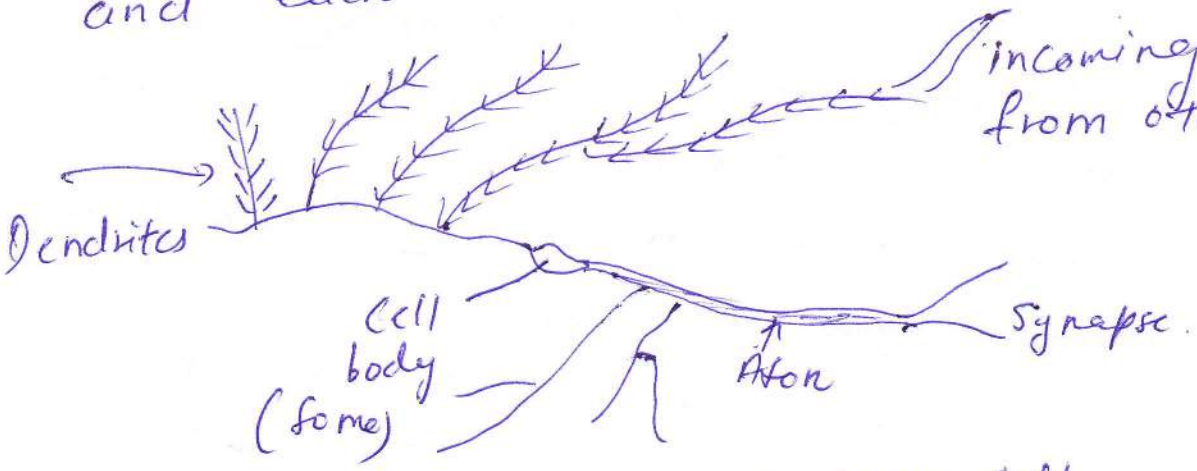
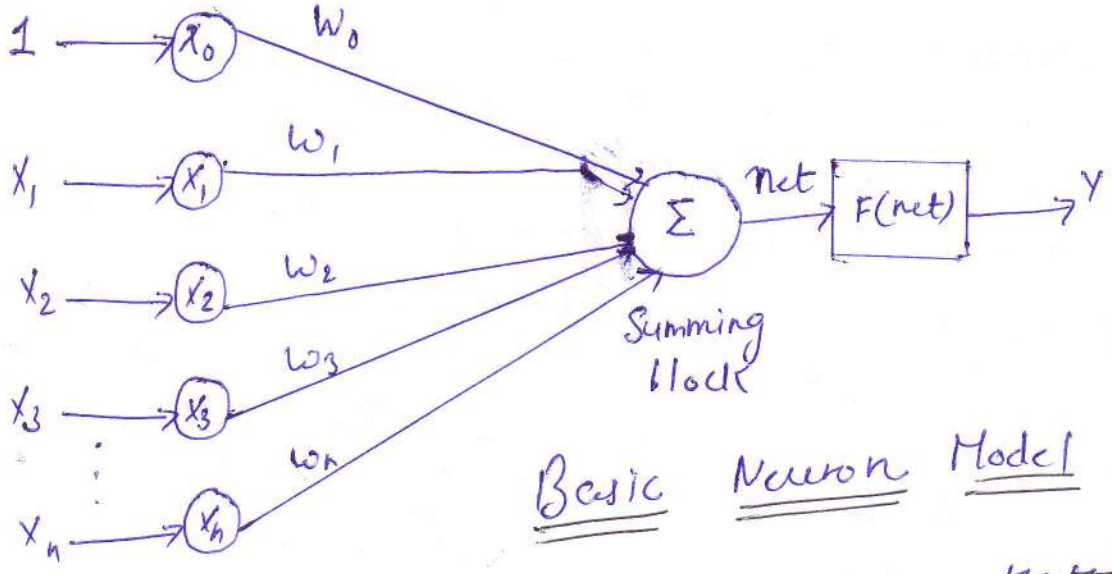


Fig 1.1 A biological Nerve cell

Neuron Concept :- terminology, Notation and representation of Neural Networks.

Artificial Neuron :- The artificial neuron (also called processing element or node) mimics the characteristics of the biological neuron. A processing element possesses a local memory and carries out localized info" processing operations. The artificial neurons has a set of 'n' inputs x_i, each represents o/p of another neuron (the subscript i takes values b/w 1 and n and indicates the source of the vector input signal). The inputs are collectively referred to as X. Each input is weighted before it reaches the main body of processing elements by the connection strength or the weight factor (or simply weight) analogous to the synaptic strength. The amount of information about the input that is required to solve a problem is stored in the form of weights. Each signal is multiplied with an associative weights $w_1, w_2, w_3 \ldots w_n$ before it is applied to the summing block. In addition the artificial neuron has a bias term w_0, a threshold value 'θ' that has to be reached or extended for the neuron to produce a signal, a nonlinear fn 'F' that acts on the produced signal 'net' and an o/p 'y' after the nonlinearity fn.

The basic model of a neuron is shown in figure 1.2. It should be noted that the input to the bias neuron is assumed to be 1.



Basic Neuron Model

The following relation describes the transfer of the basic neuron model.

$$Y = F(net)$$

where $net = w_0 + x_1 w_1 + x_2 w_2 + x_3 w_3 + \cdots + x_n w_n$

or $$net = w_0 + \sum_{i=0}^{n} x_i w_i$$

and the neuron firing condition is:

$$\left[\sum_{i=0}^{n} x_i w_i \geq 0 \quad [\text{for linear activation } f^n],\ x_0 = 1 \right]$$

or

$$\left[F(net) \geq 0 \quad [\text{for nonlinear activation } f^n] \right].$$

Architecture of A Neural Network:- The neurons are assumed to be arranged in layers and the neurons in the same layer behave in the same manner. All the neurons in a layer usually have the same activation fn. within each layer, the neurons are either fully interconnected or not connected at all. The neurons in one layer can be connected to neurons in another layer. The arrangement of neurons into layers and the connection pattern within and b/w layers is known as n/w architecture.

Input layer:- The neurons in this layer receives the external input signals and perform no computation, but simply transfer the input signals to the neurons in another layer.

output layer:- The neurons in this layer receive signals from neurons in either in the input layer or in hidden layer.

Hidden layer:- The layer of neurons that are connected inbetween the input layer and OIP layer is known as hidden layer.

Activation Functions:- The purpose of nonlinear f^n is to ensure that the neuron's response is bounded - That is, the the actual response of neuron is conditioned or clamped. as a result of small activating stimuli and thus controllable. Further inorder to achieve the advantages of multilayer nets compared with the limited capabilities of single layer NWs, nonlinear f^ns are required. Different non-linear f^ns are used, depending upon the paradigm and the algorithm used for training the network.

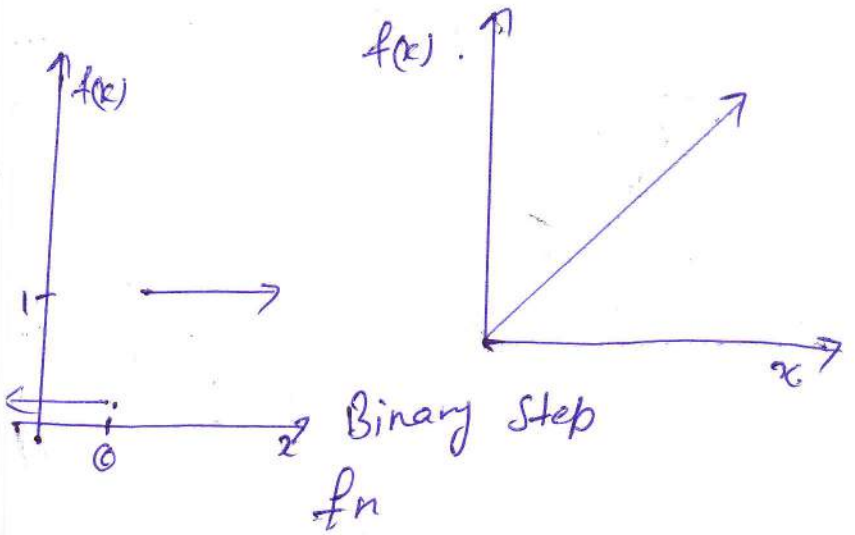
(i) **Identity Function (Linear function)**:- Identity f^n can be expressed:-

$$f(x) = x \text{ for all } x.$$

(ii) **Binary Step function**:- Binary step f^n is defined as:-

$$f(x) = \begin{cases} 1 & \text{if } x \geq \theta \\ 0 & \text{if } x < \theta \end{cases}$$

where θ is the threshold value.



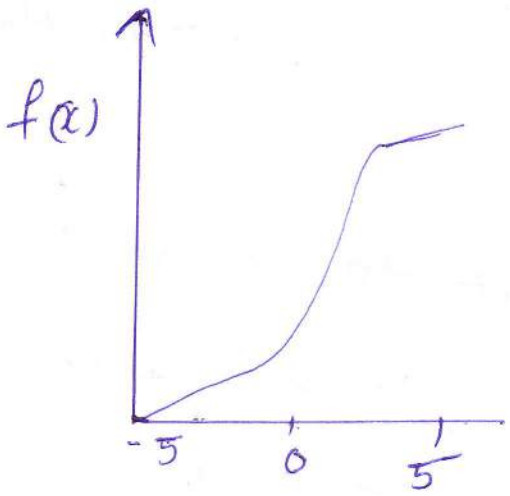
(iii) Sigmoidal function:- Sigmoidal or logistic f^n can be expressed as:

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

Sigmoidal f^n is very popular because it is monotonous, bounded & has a simple derivative

$$f'(x) = f(x)[1 - f(x)]$$

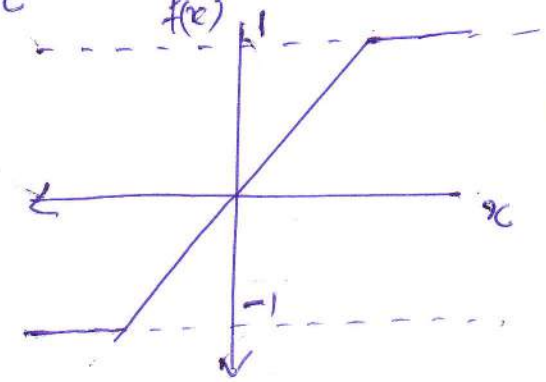
A logistic or a binary sigmoid f^n with range from 0 to 1 is shown in following fig.



Binary Sigmoidal f^n.

(iv) Bipolar Sigmoid f^n:- Bipolar sigmoid is used as an activation f^n when it reaches the desire range of o/p value is b/w -1 and +1. Bipolar activation f^n can be expressed as.

$$f(x) = \frac{2}{1+e^{x}} - 1$$

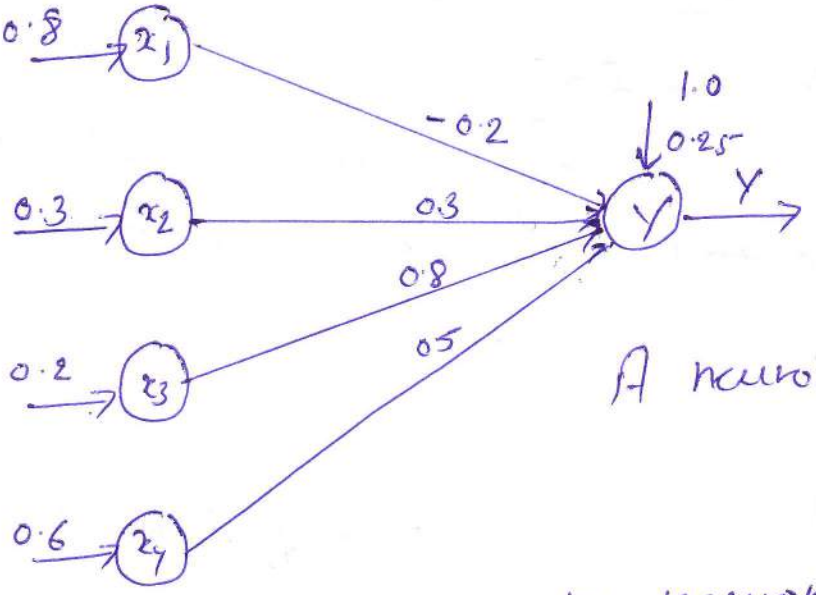


Bipolar Sigmoidal f^n.

Example:- For the network shown in figure find the o/p of the neuron Y when the activation f^n is.

(a) binary sigmoidal

(b) bipolar sigmoidal.



A neuron Model

Solution:- Net input to neuron Y is

$$y_{in} = (0.8 \times [-0.2]) + (0.3 \times 0.3) + (0.2 \times 0.8) + (0.6 \times 0.5) + 0.25 = 0.64$$

(a) For binary sigmoidal activation f^n.

$$y = f(y_{in}) = \frac{1}{1+e^{-0.64}} = 0.6548.$$

(b) For bipolar Sigmoidal activation f^n.

$$y = f(y_{in}) = \frac{2}{1+e^{0.64}} - 1 = 0.3095$$

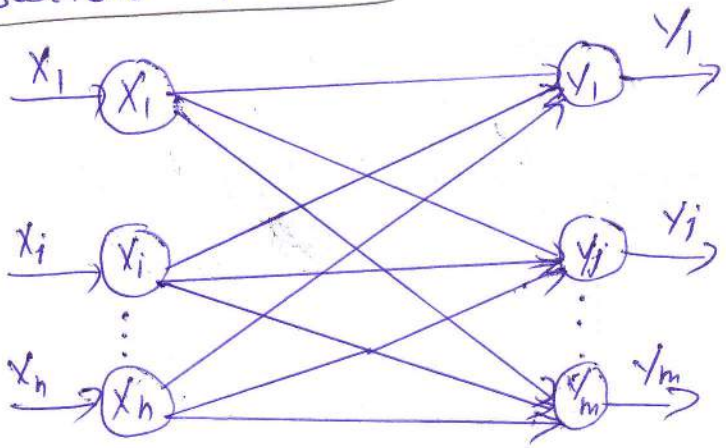
Classification of neural n/ws :- The basic building blocks of ANN are:

1) N/w architecture
2) Setting the weights
3) Activation fn

& classification are based on these building blocks.

Neural Net are often classified as single layer n/ws or multilayer n/ws. The no. of layers in a net can be classified as the no. of layers of weighted interconnn links b/w various layers. While determining the no. of layers, the input layer is not counted as a layers, becoz it does not perform any computation. The architecture of single layer & multilayer is shown as:

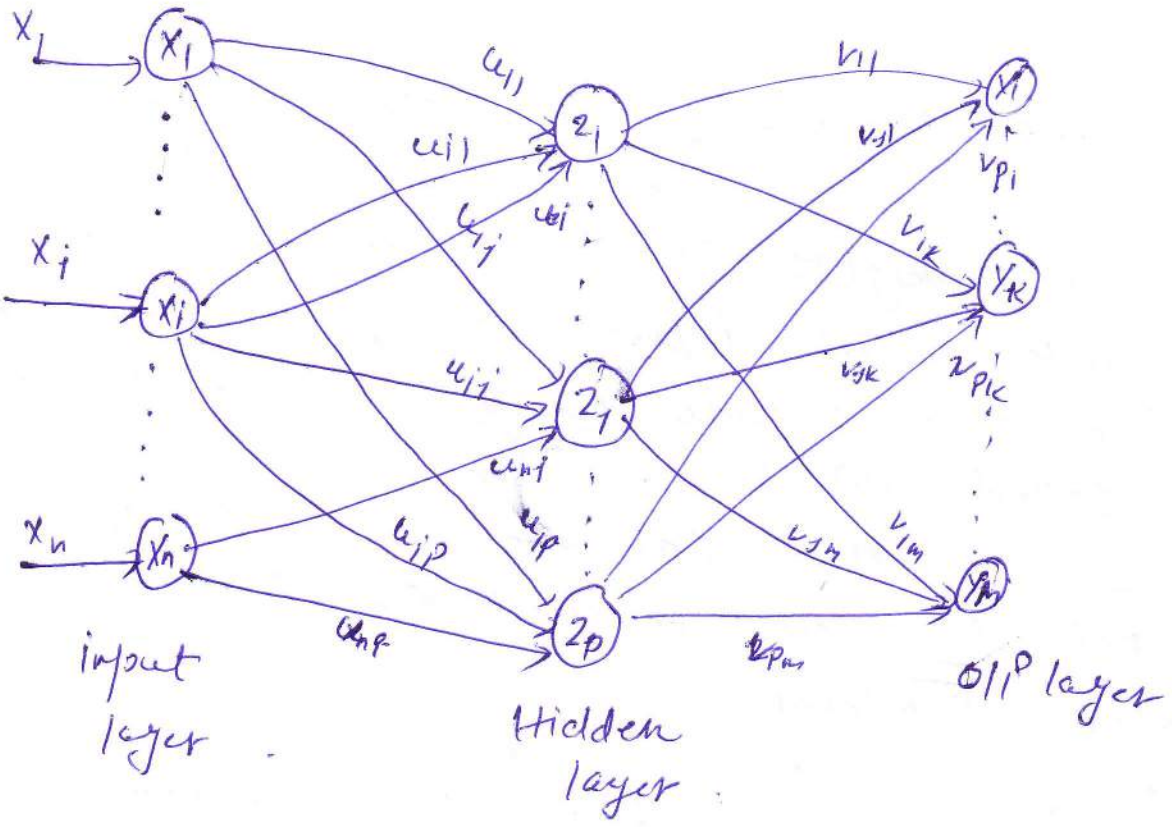
1) Single layer Network :- A single layers network consists of one layer of connection weights. The net consists of a layer of units called input layer, which receive signals from the outside world and a layer of units called o/p layer from which the response of the net can be obtained. This type of network can be used for pattern classification problems.



Single layer Neural n/w.

2) Multi layer Net: It is a feed forward N/w i.e, the net where the signals from to the o/p units in a forward direction. The multi-layer net pose one or more layers of nodes b/w the input and output units. This is advantageous over single layer net in the sense that, it can be solve more complicated problems.

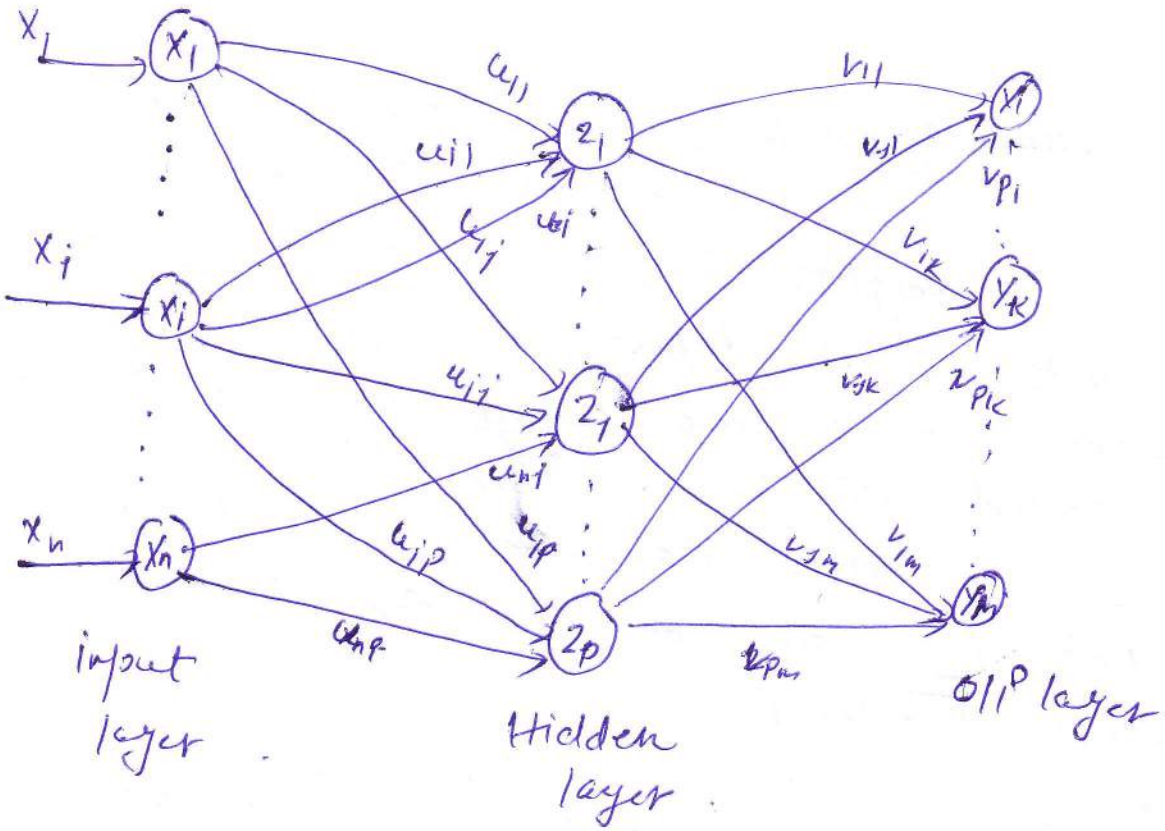
Multilayer network consist of one or more layers of units called hidden layer b/w input and o/p units. However training in multilayer N/w is complicated.



Multilayer Neural N/w.

2) Multi layer Net: It is a feed forward n/w i.e, the net where the signals from to the o/p units in a forward direction. The multi-layer net pose one or more layers of nodes b/w the input and output units. This is advantageous over single layer net in the sense that, it can be solve more complicated problems.

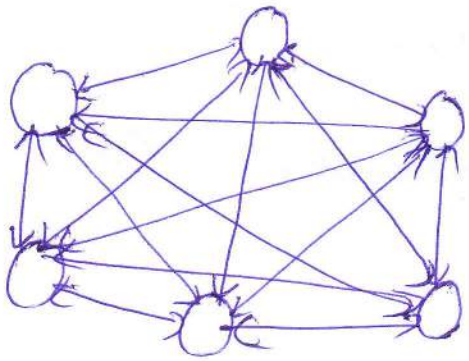
Multilayer network consist of one or more layers of units called hidden layer b/w input and o/p units. However training in multilayer n/w is complicated.



Multilayer Neural n/w.

3) Fully Recurrent N/W :- The fully recurrent network is perhaps the simplest of neural N/W architecture. All units are connected to all other units. and every unit is both an input and an O/P. Typically a set of pattern is instantiated on all of the units, one at a time. As each pattern is initiated the weights are modified.

Recurrent N/Ws are also useful in that they allow N/Ws to process sequencial info".



Fully Recurrent N/W.

Training an ANN (Artificial Neural N/W) : The most important characteristic of an artificial Neural N/W is its ability to learn. Learning in both biological and artificial neural N/Ws is undergoing intense research. The basic questions regarding the human learning process are : How do we learn? Which is the most efficient process for learning? How much and how fast can we learn?

Generally speaking, learning is a process by which a neural N/W adapts itself to a stimulus by making Parameter adjustments and and producing a desired response.

learning is also is continuous process of the o/p stimuli. If the input stimuli. of the input stimuli is not recognized by the n/w, then the n/w develops a new classification.

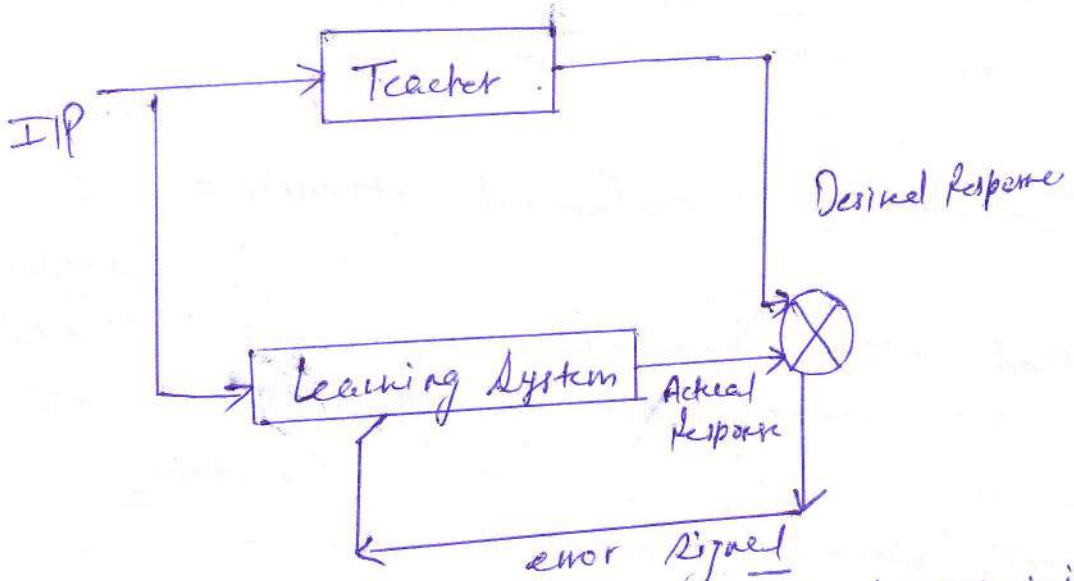
Learning (training) is a process (of o/p stimuli) in which the n/w adjusts its parameters the (synaptic weights) in response to input stimuli so that the actual o/p response converges to desired o/p response. When the actual o/p response is the same as the desired one, the n/w has completed the learning phase and the n/w has acquired knowledge

Learning or training algorithms can be categorized as:

1). Supervised training
2). Unsupervised training
3). Reinforced training.

Supervised training: Supervised training requires the pairing of each I/P vector with a target vector represented the desired o/p. The I/P vector together with the corresponding target vector is called training pair. During the training session an input vector is applied to the n/w. and it results in an o/p vector. This response is compared with the target response. If the actual response differs from the target response, the n/w generate an error signal. This error signal is then used to calculate the adjustment that should be made in the

Synaptic weights so that the actual O/P matches the target O/P. The error minimization in this kind of training requires a supervisor or a teacher, hence the name supervised training. In artificial neural n/ws, The calculation that is required to minimize errors depends on the algorithm used, which is normally based on optimization techniques. Figure shows the supervised training method. Supervised training methods is used in to perform nonlinear mapping in pattern classification nets, pattern association nets and multilayer neural nets.



Block Diagram of Supervised Training.

Unsupervised Training: Unsupervised training is employed in self organizing neural nets. In contrast to supervised learning unsupervised training does not require a teacher. In this method of training, the input vectors of similar types are grouped without the use of training data to specify

how a typical member of each groups looks or to which group a member belongs. During training the neural N/W receive I/P Signals (patterns) and organizes their pattern into categories. Later new I/P Pattern is applied, the neural N/W provides an O/P response indicating the class to which the I/P pattern belongs. If a class can't be found for I/P pattern, a new class is generated.

Even though unsupervised training does not require a teacher, it requires certain guidelines to form groups. Grouping can be done based on colour, shape or any other property of object. If no guidelines are given grouping may or may not be successful.

Reinforced Training:- Reinforced training is similar to supervised training. In this method, the teacher does not indicate how close is the actual O/P to the desired O/P but yields only a pass or fail indication. Thus, the error signal generated during reinforced training is binary.

Hebb learning Rule:- "Increase the value of weight if the O/P is active when the input associated with that weight is also active"

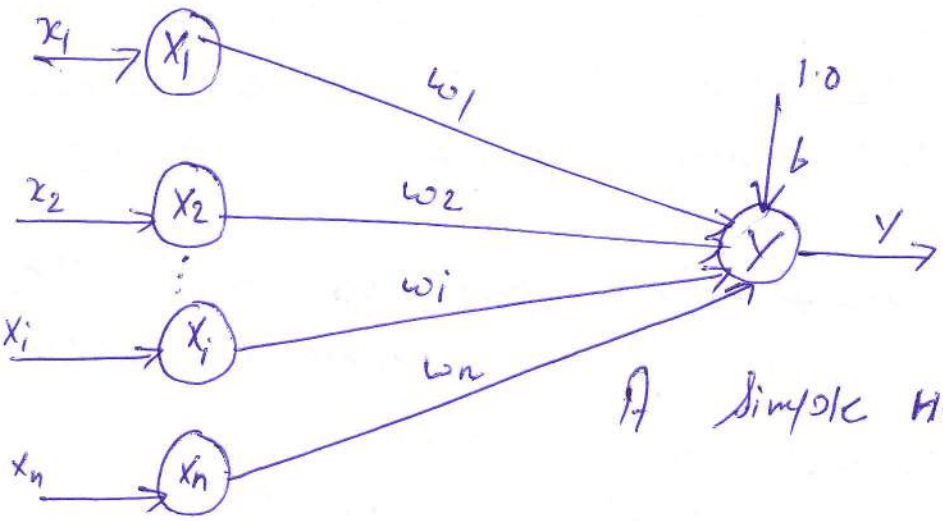
If we assume that the input x and the O/P y has binary values, the weight value will be incremented only when the O/P is '1' & the corresponding I/P is '1'.

Slent dephased the hebb's hypothesis as

1). If two neurons on either side of a synapse (connection) are activated simultaneously, Then the strength of that synapse is selectively asynchronously., Then synapse is selectively weakened or eliminated.

Hebb net:- Based on Hebbian learning rule, McClelland and Rumelhart developed the Hebb net. This net consists of a Single layer feedforward neural N/W. This net has fixed bias value of 1: The I/P and the O/P patterns must be of bipolar form.

The Hebb net does not learn if the I/P - O/P patterns are in binary form. This Hebb net does not learn if the I/P - O/P patterns are in binary form. This is extreme limitation of Hebb net.



A Simple Hebb net

Algorithm to train Hebb net:-

Step 1: Initialize all weights to zero, $w_i = 0$ for $(i = 1, 2, 3, \ldots, n)$ where n is the no. of I/P neurons.
Initialize the bias value to zero, $b = 0$

Step 2: For each I/P vector, target O/P pair s.t do steps 3 to 6

Step 3: Set activations for input units with the I/P vector.

$$x_i = s_i \quad (\text{for } i = 1, 2, 3 \ldots, n)$$

Step 4: Set the corresponding O/P value to the O/P neuron

$$Y = t$$

Step 5: Apply Hebb rule and modify weights

$$w_i(\text{new}) = w_i(\text{old}) + x_i y \quad (\text{for } i = 1, 2, 3, \ldots n)$$

Step 6: Adjust the bias.

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

The about algorithm requires only one pass through the training set. The weight update can be written in the vector form as:

$$W_{(new)} = W_{(old)} + X^T y$$

Note:
W is a column vector.
X is row vector.
y is a scalar.

Example: Design a Hebb net to implement logical AND fn using bipolar input-o/p patterns

The following example is illustrated below:-

Soln. The training patterns for an AND logic fn is shown below.

Training Patterns.

	I/P x_1	I/P x_2	b	Target y
X_1	-1	-1	1	y_1 -1
X_2	-1	1	1	y_2 -1
X_3	1	-1	1	y_3 -1
X_4	1	1	1	y_4 1

A single layer network with 2 input neurons, one bias and one o/p neuron is considered.
The intial weight are set to zero.

$$W_{(old)} = [0 \ 0 \ 0]^T$$

Case 1:
For the first I/P $X_1 = [-1 \ -1 \ 1]$ and the target $Y_1 = [-1]$ the updated weight is:

$$W_{(new)} = W_{(old)} + X_1^T Y_1$$
$$= [0 \ 0 \ 0]^T + [-1 \ -1 \ 1]^T [-1]$$
$$= [0 \ 0 \ 0]^T + [1 \ 1 \ -1]^T = [1 \ 1 \ -1]^T$$

Case II:

On applying the second input vector $X_2 = [-1 \ 1 \ 1]$ and the corresponding target $Y_2 = [-1]$, the new weight vector is:-

$$W_{(new)} = W_{(old)} + X_2^T Y_2.$$

$$= [1 \ 1 \ -1]^T + [-1 \ 1 \ 1]^T [-1]$$

$$= [1 \ 1 \ -1]^T + [1 \ -1 \ -1]^T$$

$$= [2 \ 0 \ -2]^T$$

Case III: For the third input pattern $X_3 = [1 \ -1 \ 1]$ and the corresponed target $Y_3 = [-1]$ the new weight vector is.

$$W_{(new)} = W_{old} + X_3^T Y_3$$

$$= [2 \ 0 \ -2]^T + [1 \ -1 \ 1]^T [-1]$$

$$= [2 \ 0 \ -2]^T + [-1 \ 1 \ -1]^T$$

$$= [1 \ 1 \ -3]^T$$

Case IV

Applying the fourth input pattern $X_4 = [1 \ 1 \ 1]$ and the corresponding target $Y_4 = [1]$ the final weight vector is.

$$W_{(new)} = W_{(old)} + X_4^T Y_4$$

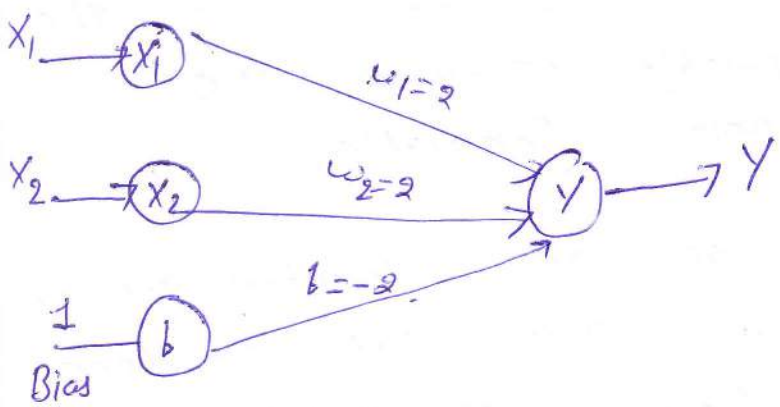
$$= [1 \ 1 \ -3]^T + [1 \ 1 \ 1]^T [1]$$

$$= [1 \ 1 \ -3]^T + [1 \ 1 \ 1]^T$$

$$= [2 \ 2 \ -2]^T$$

The Hebb net architecture with the final weight vector is shown in fig.

Hebb net for AND fn.



Linear Separability:- The intension of training a network is to find the weights that will respond correctly when the net is presented with an input pattern that is similar to one of the training patterns.

The activation fn of o/p neuron is a step fn. Consider a sample neural net with n input neurons and one o/p neuron. If the neuron response of the o/p neuron is one (1) for an input pattern then the input pattern is a member of the class. If the respone of o/p is -1 then the input does not belong to the class.

The net input to the o/p neurons can be written as $y_{in} = w_0 + \sum x_i w_i$

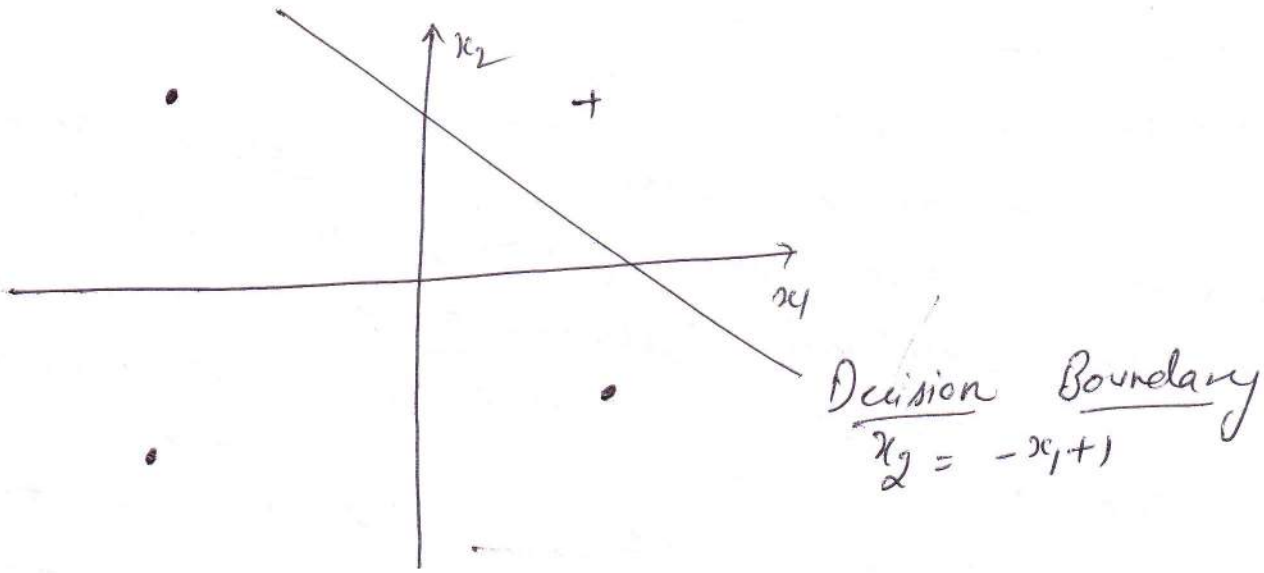
Using the above equation, the decision boundary b/w the region when $y_{in} > 0$ and $y_{in} < 0$ can be determined. Depending on the no. of input neurons in the n/w, this eqn represents a line or a place or a hyper plane.

If it is possible to find the weights so that all of the training input vectors for which the correct response is 1 lie on

The ~~the~~ either side of the boundary, Then the problem is linearly Separable otherwise the problem is linearly non-Separable.

The given AND f^n Can be represented as a classification problem b/w inputs that produces a '1' as o/p and those that produces '-1' as o/p. The f^n Can be represented by means of pattern space in which the inputs represents the coordinate system for that space and set of co-ordinates gives the position of the feature in that space.

Fig: shows the two input variables labelled x_1 and x_2 on the pattern space diagram. The four points $(-1,-1)$, $(-1,1)$, $(1,-1)$ and $(1,1)$ represents the position in the pattern space. At each point, the value of the o/p of f^n, $y = x_1$ AND x_2 is marked as · (dot) for -1 or a cross for (+1)



In this example, we have considered the Separation of the input space into regions where the response is -ve. The value of x_1 and x_2 for which the net gives a +ve response and the values for which it gives a -ve response is known as the separating line,

$$x_1 w_1 + x_2 w_2 + b = 0$$

using the final weight value, the eqn for the boundary line is $2x_1 + 2x_2 - 2 = 0$ or $x_2 = -x_1 + 1$.
The straight line $x_2 = -x_1 + 1$, Seperate the pattern space into two regions s.t for unit response the point lie on one side of the boundary while for all the other three input patterns $[(-1,-1), (-1,1), (1,-1)]$ for which the o/p response is '-1', the points lie on the other side of boundary. Since the straight line separates the two classes, this problem is referred to as a linearly Separable problem.

Note: In general, for n input M/Cs, the relation that determines the decision boundary is given by

$$b + \sum_{i=1}^{n} x_i w_i = 0$$

Depending upon the no. of o/p neurons in the M/C, the eqn with represent either a line, plane or a hyper-plane.

If There are weights and a bias so That all of The training input vectors for which The Correct response is '+1' lie on one side of the decision boundary and all of The training input vectors for which the Correct response is '-1' lie on other side of the decision boundary, Then the problem is linearly Separable.

Consider the another problem to define XOR f^n.

By apply Hebb rule method to training the patterns That define XOR f^n & find the Soln.

Soln: The training patterns for an XOR logic f^n is shown as:

	I/P x_1	x_2	b	Target y
X_1	-1	-1	1	Y_1 -1
X_2	-1	1	1	Y_2 1
X_3	1	-1	1	Y_3 1
X_4	1	1	1	Y_4 -1

A Single layer N/W with 2 input neurons, one bias and one O/P neuron is Considered.

The initial weight are set to zero.

$$W_{(old)} = [0\ 0\ 0]^T$$

Case I:
For the first input vector $X_1 = [-1 \ -1 \ 1]$ and the target $Y_1 = [-1]$ the updated weight is

$$W_{(new)} = W_{(old)} + X_1^T Y_1$$
$$= [0 \ 0 \ 0]^T + [-1 \ -1 \ 1]^T [-1]$$
$$= [0 \ 0 \ 0]^T + [1 \ 1 \ -1]^T$$
$$= [1 \ 1 \ -1]^T$$

Case II: on applying the second input vector $X_2 = [-1 \ 1]$ and the corresponding weight target $Y_2 = [1]$ the new weight vector is.

$$W_{(new)} = W_{old} + X_2^T Y_2$$
$$= [1 \ 1 \ -1]^T + [-1 \ 1 \ 1]^T [1]$$
$$= [1 \ 1 \ -1]^T + [-1 \ 1 \ 1]^T$$
$$= [0 \ 2 \ 0]^T$$

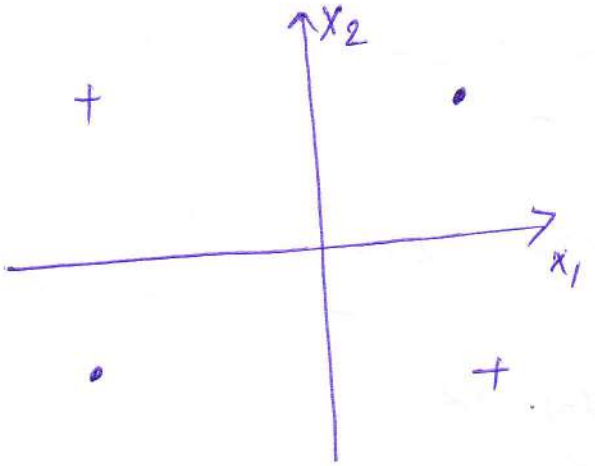
Case 3: For third input pattern $X_3 = [1 \ -1 \ 1]$ and the corresponding target $Y_3 = [1]$ the new weight vector is.

$$W_{(new)} = W_{(old)} + X_3^T Y_3$$
$$= [0 \ 2 \ 0]^T + [1 \ -1 \ 1]^T [1]$$
$$= [0 \ 2 \ 0]^T + [1 \ -1 \ 1]^T$$
$$= [1 \ 1 \ 1]^T$$

Case iv:

$$W_{(new)} = W_{(old)} + X_4^T Y_4$$
$$= [1 \ 1 \ 1]^T + [1 \ 1 \ 1]^T [-1]$$
$$= [1 \ 1 \ 1]^T + [-1 \ -1 \ -1]^T$$
$$= [0 \ 0 \ 0]^T$$

The final weight do not give the correct o/p for all input patterns. Fig. shows that the input patterns are linearly non-separable. The XOR f^n is classification example or problem that is not linearly separable.



Example:- Design or develop a Hebb net to implement logical OR f^n using bipolar I/P - O/P patterns. The following example is as:-

Soln:- The training patterns for logical OR f^n is shown below.

Training patterns.

	I/P x_1	x_2	b	Target y
X_1	-1	-1	1	Y_1 -1
X_2	-1	1	1	Y_2 1
X_3	1	-1	1	Y_3 1
X_4	1	1	1	Y_4 1

The initial weights are set to zero.

$w(old) = [0 \quad 0 \quad 0]^T$

Cycle 1:- For the first input $X_1 = [-1 \quad -1 \quad 1]$ and the target, $Y_1 = [-1]$ the updated weight is:-

$$w_{(new)} = w_{(old)} + x_1^T y_1$$

$$= [0 \ 0 \ 0]^T + [-1 \ -1 \ 1]^T [-1]$$

$$= [0 \ 0 \ 0]^T + [1 \ 1 \ -1]^T$$

$$= [1 \ 1 \ -1]^T$$

Case II: on applying the second input vector $X_2 = [-1 \ 1 \ 1]$ and the corresponding target, $Y_2 = [1]$ the new weight vector is:

$$w_{(new)} = w_{(old)} + x_2^T y_2$$

$$= [1 \ 1 \ -1]^T + [-1 \ 1 \ 1]^T [1]$$

$$= [1 \ 1 \ -1]^T + [-1 \ 1 \ 1]^T$$

$$= [0 \ 2 \ 0]^T$$

Case iii: For the third input pattern $X_3 = [1 \ -1 \ 1]$ and the corresponding target $Y_3 = [1]$ the new weight vector is:

$$w_{(new)} = w_{(old)} + x_3^T y_3$$

$$= [0 \ 2 \ 0]^T + [1 \ -1 \ 1]^T [1]$$

$$= [0 \ 2 \ 0]^T + [1 \ -1 \ 1]^T$$

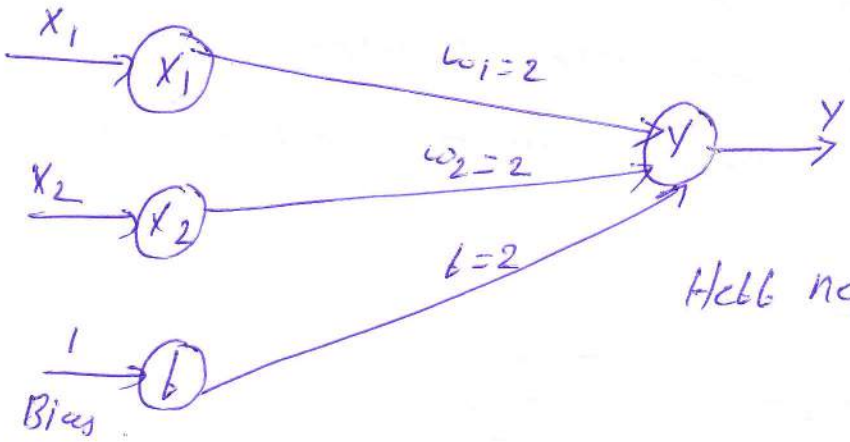
$$= [1 \ 1 \ 1]^T$$

Case IV: Applying the fourth input pattern $X_4 = [1 \ 1 \ 1]$ and the corresponding target $Y_4 = [1]$ the final weight vector is:

$$w_{(new)} = w_{(old)} + x_4^T y_4$$

$$= [1 \quad 1 \quad 1]^T + [1 \quad 1 \quad 1]^T [1]$$
$$= [1 \quad 1 \quad 1]^T + [1 \quad 1 \quad 1]^T$$
$$= [2 \quad 2 \quad 2]^T$$

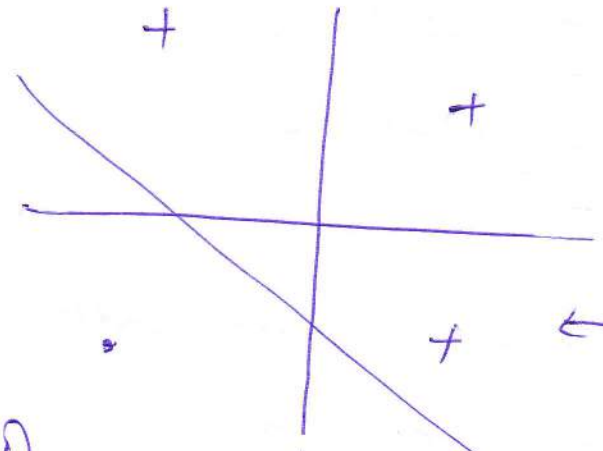
The final weights are marked in the Hebb net and shown in fig:



Hebb net for OR function

using the final weight the eqn for the boundary line is $2x_1 + 2x_2 + 2 = 0$ or $x_2 = -x_1 - 1$

The decision region for the net is shown in fig. From fig it can be observe that the straight line $x_2 = -x_1 - 1$, seperates the pattern space into two regions.

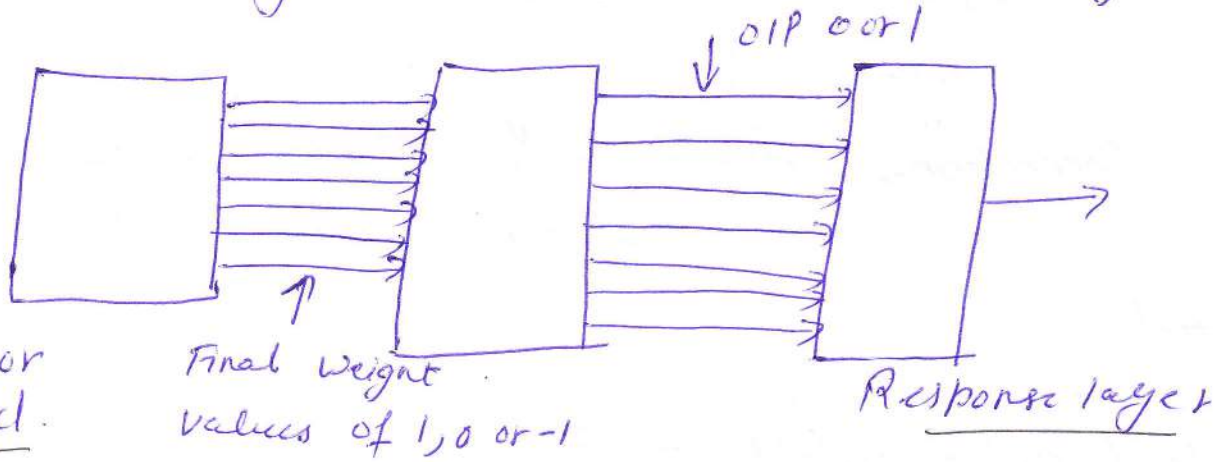


Decision boundary for OR fn

The input pattern [1,1], (1,-1) (1,1)] for which the o/p response is '1' lie on one side of boundary and the input pattern (-1,-1) for which the o/p response is (-1) lie on other side of the boundary.

Representation of Perception and issues:

Perception The first adaptive n/w architecture was invented by Frank Rosenblatt in 1957, Rosenblatt's research was more oriented towards modeling The brain in an attempt to understand memory, learning and cognitive processes. The model Perception model developed by Rosenblatt is shown in fig.



Block model employs binary activations for the sensory and associatory units. and an activation of 1, 0 or -1 for the response units, The signal sent from associator units to o/p unit is a binary signal (0 or 1)

The activation fn for the response unit is:

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

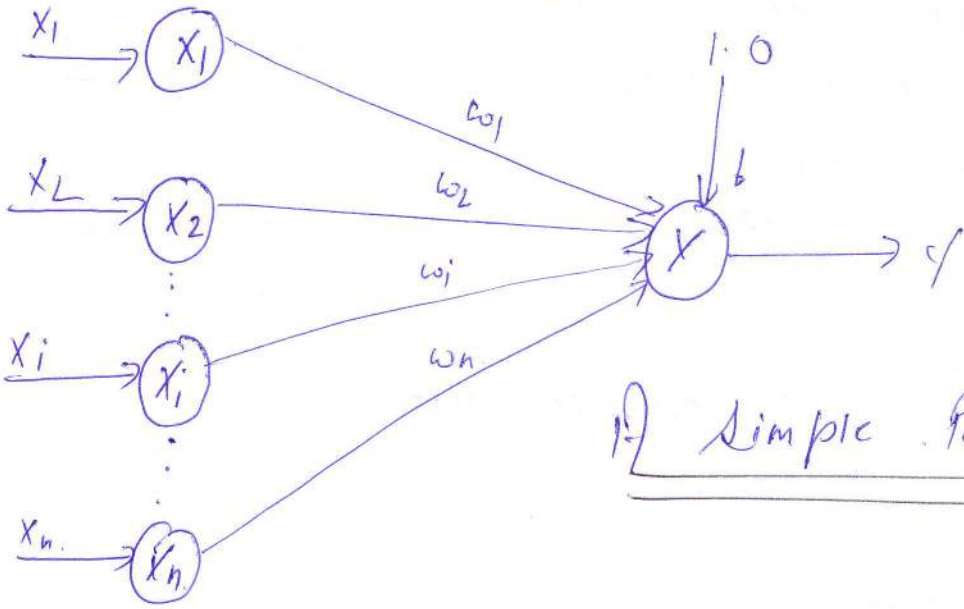
$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \le y_{in} \le \theta \\ -1 & \text{if } y_{in} < \theta \end{cases}$$

The weights that are connected b/w the Association layer and response layer are adjusted during training. After presenting each training input vector, the net calculates the net input to the perceptron and then the o/p response of the perceptron. The o/p signals are compared with the target value whether an error has occured. If an error has occured, then only the weights on the connections from units that sent a non-zero signals to the o/p unit will be adjusted.

i.e

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

where t is target value 1 or -1 and α is the learning rate.



1) Simple Perceptron

Perceptron Learning Algorithm:- This algorithm is suitable for binary/bipolar input vectors with bipolar targets.

Step 1:- Initialize weights and bias. For simplicity assume zero values. Set learning rate α ($0 < \alpha \leq 1$)

Step 2: While **Stopping Condition** is false, do Steps 3 to 7

Step 3: For each **training pair** s:t, do steps 4 to 6.

Step 4: Set the input activations.

$$x_i = s_i$$

Step 5: Compute net input to the **perceptron** and the O/P response of the perceptron

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

$$y = \begin{cases} 1 & y_{in} > \theta \\ 0 & -\theta \leq y_{in} \leq \theta \\ -1 & y_{in} < \theta \end{cases}$$

Step 6: Update the bias and weights if the target is not equal to the O/P response

if $t \neq y$

if $x_i \neq 0$

$w_i(new) = w_i(old) + \alpha x_i t \quad i = 1, 2, 3 \cdots n$

else no change in weights.

$b(new) = b(old) + \alpha t$.

Step 7: Test for **Stopping Condition**. If no weight change in Step 3 stop else continue

Note: 1) Weights Connecting active input units are updated ie $x_i \neq 0$

2) Weights are updating for pattern that do not produce the Correct O/P value.

Perceptron Application Algorithm:-

Step 1: Apply Perceptron training Algorithm to train the weights.

Step 2: For each input vector x to be Classified, do Step 3 to 4.

Step 3: Set activation of input units.

Step 4: Compute net input & O/P of Perceptron.

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

$$y = \begin{cases} 1 & y_{in} > \theta \\ 0 & -\theta \leq y_{in} \leq \theta \\ -1 & y_{in} < -\theta \end{cases}$$

Example: Develop a Perceptron n/w to implement an AND fn.

Solution:- For the First Input Sample $x_0 = 1$, $x_1 = 1$; $x_2 = 1$; $t = 1$, The net input to the O/P neuron y_{in} is:

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$$y_{in} = 0 + (1 \times 0) + (1 \times 0)$$

$$y_{in} = 0$$

Applying the activation fn.

$$y = \begin{cases} 1 & y_{inj} > \theta \\ 0 & -\theta \le y_{in} \le \theta \\ -1 & y_{inj} < -\theta \end{cases}$$

The o/p of the perceptron, $y = 0$

Since $y \neq t$, the new weights are calculated as:

$$w_0(new) = w_0(old) + \alpha \cdot t$$
$$= 0 + (1 \times 1) = 1$$
$$w_i(new) = w_i(old) + \alpha x_i t$$
$$w_1(new) = w_1(old) + \alpha x_1 t$$
$$= 0 + (1 \times 1 \times 1) = 1$$
$$w_2(new) = w_2(old) + \alpha x_2 t$$
$$= 0 + (1 \times 1 \times 1) = 1$$

The weight matrix after presenting the first sample, $W = [1\ 1\ 1]$

for the second input sample $x_0 = 1$, $x_1 = -1$, $x_2 = 1$ $t = -1$, the net input to the o/p neuron y_{in} is:

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$$y_{in} = 1 + (-1 \times 1) + (1 \times 1)$$

$$y_{in} = 1$$

The o/p of the perceptron is $y = 1$

Since $y \neq t$, the new weights are calculated as:

$$w_0(new) = w_0(old) + \alpha t$$
$$= 1 + (1 \times 1) = 0$$
$$w_i(new) = w_i(old) + \alpha x_i t$$
$$w_1(new) = w_1(old) + \alpha x_1 t$$
$$= 1 + (1 \times -1 \times -1) = 2$$
$$w_2(new) = w_2(old) + \alpha x_2 t$$
$$= 1 + (1 \times 1 \times -1) = 0$$

The weight matrix after presenting the second sample $w = [0\ 2\ 0]$.

For third input sample $x_0 = 1$; $x_1 = 1$; $x_2 = -1$; $t = -1$

The net input to the O/P neuron, y_{in} is

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$
$$y_{in} = 0 + (1 \times 2) + (-1 \times 0)$$
$$y_{in} = 2$$

The O/P of the perceptron is $y = 1$

Since $y \neq t$, the new weights are calculated.

$$w_0(new) = w_0(old) + \alpha t$$
$$= 0 + (1 \times -1) = -1$$
$$w_i(new) = w_i(old) + \alpha x_i t$$
$$w_1(new) = w_1(old) + \alpha x_1 t$$
$$= 2 + (1 \times 1 \times -1) = 1$$
$$w_2(new) = w_2(old) + \alpha x_2 t$$
$$= 0 + (1 \times -1 \times -1) = 1$$

The weight matrix after presenting the second sample, $w = [-1\ 2\ 1]$

For Fourth input Sample $x_0 = 1$, $x_1 = -1$, $x_2 = -1$; $t = -1$, The next net input to o/p neuron.

y_{in} is:

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$$y_{in} = -1 + (-1 \times 1) + (-1 \times 1)$$

$$y_{in} = -3$$

The o/p of the perceptron is $y = -1$

Since $y = t$, no change in weight occurs.

The weight matrix after presenting the fourth Sample $W = [-1 \ 1 \ 1]$

Example:- Develop a Perceptron n/w to implement an OR f^n.

Solution:- Let the initial weights be Zero.

For the first input Sample $x_0 = 1$, $x_1 = 1$, $x_2 = 1$, $t = 1$ the net input to the o/p neuron

y_{in} is.

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$$y_{in} = 0 + (1 \times 0) + (1 \times 0)$$

$$y_{in} = 0.$$

Applying the activation f^n.

$$y = \begin{cases} 1 & y_{inj} > 0 \\ 0 & -0 \leq y_{in} \leq 0 \\ -1 & y_{inj} < -0 \end{cases}$$

The o/p of the Perceptron $y = 0$

Since $y \neq t$ The new weights are Calculated as.

$$w_0(new) = w_0(old) + \alpha t$$
$$= 0 + (1 \times 1) = 1$$
$$w_i(new) = w_i(old) + \alpha x_i t$$
$$w_1(new) = w_1(old) + \alpha x_1 t$$
$$= 0 + (1 \times 1 \times 1) = 1$$
$$w_2(new) = w_2(old + \alpha x_2 t$$
$$= 0 + (1 \times 1 \times 1) = 1$$

The weight matrix after presenting the first sample, $w = [1 \quad 1 \quad 1]$

The second matrix after presenting the first sample for $x_0 = 1$, $x_1 = -1$, $x_2 = 1$, $t = 1$ the net input to the o/p neuron y_{in} is:

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$$y_{in} = 0 + (-1 \times 1) + (1 \times 1)$$

$$y_{in} = 1$$

The o/p of the Perceptron is $y = 1$

Since $y = t$, no change in weight occurs

For third input sample $x_0 = 1$; $x_1 = 1$, $x_2 = -1$ and $t = 1$, the net input to the o/p neuron y_{in} is:

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$y_{in} = 1 * (1 \times 1) + (-1 \times 1)$

$y_{in} = 1$

The o/p of the perceptron is $y = 1$

Since $y = t$, no change in weight occurs.

The weight matrix after presenting the fourth sample $W = [1\ 1\ 1]$

Example:

Develop a Perceptron N/W to implement AND NOT fn.

Soln: For first input sample $x_0 = 1$, $x_1 = 1$, $x_2 = 1$ $t = -1$, the net o/p neuron, y_{in} is

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$$y_{in} = 0 + (1 \times 0) + (1 \times 0)$$

$$y_{in} = 0.$$

Applying the activation fn.

$$y = \begin{cases} 1 & y_{in} \geq 0 \\ 0 & -0 \leq y_{in} \leq 0 \\ -1 & y_{in} \leq -0 \end{cases}$$

The o/p of the Perceptron $y = 0$

Since $y \neq t$, the new weights are calculated as

$$w_0(new) = w_0(old) + \alpha t$$

$$= 0 + (1 \times 1) = 1$$

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

$w_1(new) = w_1(old) + \alpha x_1 t$

$= 0 + (1 \times 1 \times -1) = -1$

$w_2(new) = w_2(old) + \alpha x_2 t$

$= 0 + (1 \times 1 \times -1) = -1$

The weight matrix after presenting the first sample, $w = [1, -1, -1]$

For second input sample, $x_0 = 1$, $x_1 = -1$, $x_2 = 1$ $t = -1$, the net I/P to o/p neuron y_{in} is.

$$y_{in} = w_0 + \sum_{i=1}^{h} x_i w_i$$

$y_{in} = 1 + 1 - 1 = 1$

$y = 1$

The o/p of the perceptron is $y = 1$

Since $y \neq t$, the new weights are calculated as:

$w_0(new) = w_0(old) + \alpha t$

$= 1 + (1 \times -1) = 0$

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

$w_1(new) = w_1(old) + \alpha x_1 t$

$-1 + (1 \times -1 \times -1) = 0$

$w_2(new) = w_2(old) + \alpha x_2 t$

$= -1 + (1 \times 1 \times -1) = -2$

The weight matrix after presenting the Second Sample $W = [0 \;\; 0 \;\; -2]$

For Third input sample $x_0 = 1$, $x_1 = 1$, $x_2 = -1$, $t = 1$ the net input to the O/P neuron y_{in} is:

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$$y_{in} = 0 + (1 \times 0 + -1 \times -2)$$

$$y_{in} = -2.$$

The o/p of the perceptron is $y = +1$

Since $y = t$, no change in wight occurs.

For fourth input Sample $x_0 = 1$, $x_1 = -1$, $x_2 = -1$ $t = -1$, The net input to the o/p neuron y_{in} is.

$$y_{in} = w_0 + \sum_{i=1}^{n} x_i w_i$$

$$y_{in} = 0 + (-1 \times 0) + (-1 \times -2)$$

$$y_{in} = 2.$$

The O/P of the perceptron is $y = 1$

Since $y \neq t$, The new weights an Calculated as:

$$w_0(new) = w_0(old) + \alpha t$$
$$= 0 + (1 \times -1) = -1$$

$$w_i(new) = w_1(old) + \alpha x_i t$$
$$= 0 + (1 \times -1 \times -1) = 1$$

$$w_2(new) = w_2(old) + \alpha x_2 t$$
$$-2 + (1 \times -1 \times -1) = -1$$

The weight matrix after presenting the fourth Sample $W = [-1 \;\; 1 \;\; -1]$