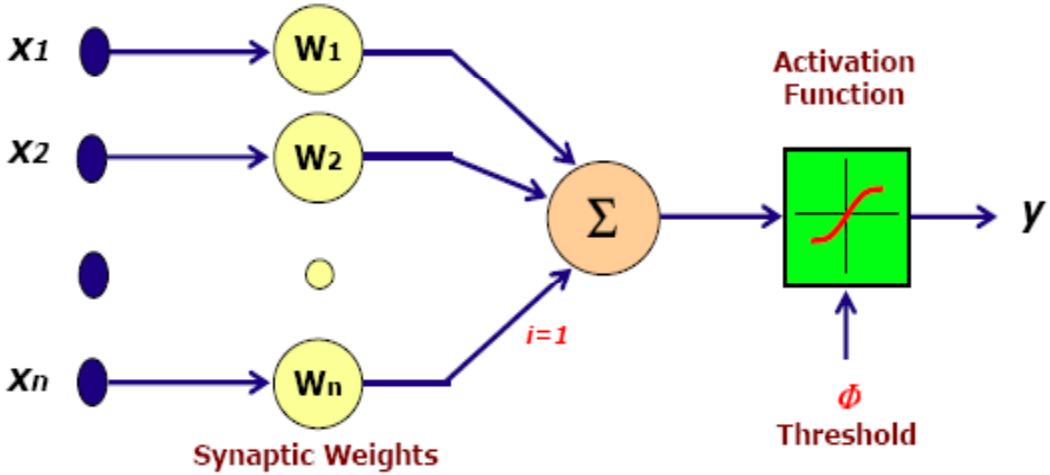
Basic Elements of Artificial neuron net

- Neuron consists of three basic components: weight, threshold and a single activation function.



- **Weight:** the value w_1, w_2 w_3 ------ w_n are weights to determine the strength of input vector $X = [x_1, x_2, x_3, \text{------} x_n]^T$. Each input is multiplied by the associated weight of the neuron connection X^TW. The +ve weight excites and the –ve weight inhibits the node output.

$$I = X^T.W = x_1 w_1 + x_2 w_2 + ... + x_n w_n = \sum_{i=1}^{n} x_i w_i$$

Basic Elements of Artificial neuron net

- **Threshold:** The node's internal threshold Φ is the magnitude offset. It affects the activation of the node Y as:

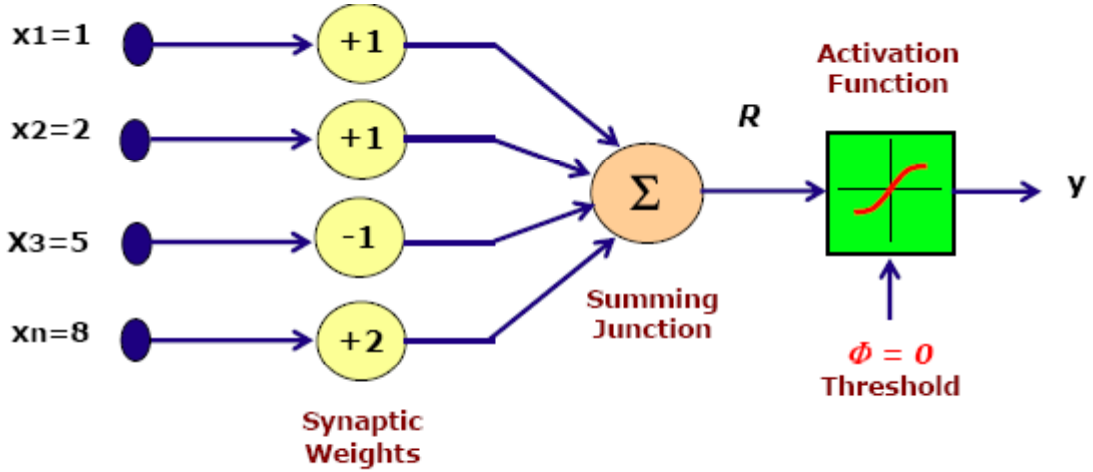
$$Y = f(I) = f\left\{ \sum_{i=1}^{n} x_i w_i - \Phi_k \right\}$$

- To generate the final output Y, the sum is passed on to a non-linear filter f called activation function or transfer function which releases the output Y

- **In, practice, neuron generally do not fire (produce an output) unless their total input goes above a threshold value.**

Example

- A Neural Network consists 4 inputs with the weights as shown below.



The output R of the network, prior to activation function stage is

$$R = W^T . X = \begin{bmatrix} 1 & 1 & -1 & 2 \end{bmatrix} \bullet \begin{pmatrix} 1 \\ 2 \\ 5 \\ 8 \end{pmatrix} = 14$$

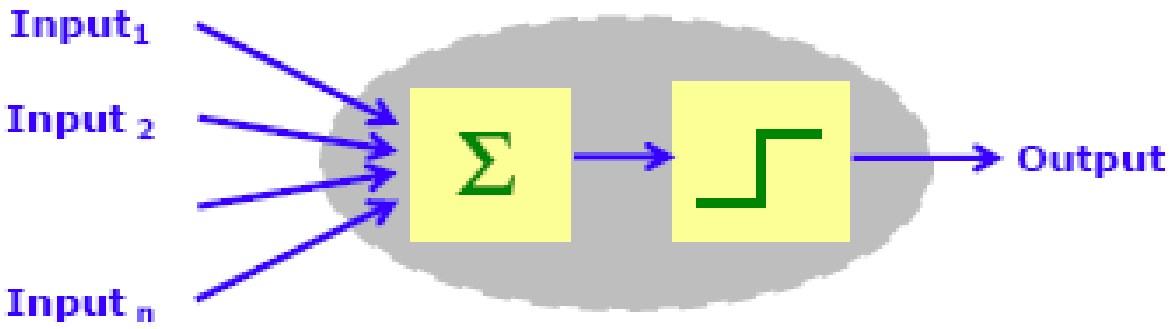
$$= (1 \times 1) + (1 \times 2) + (-1 \times 5) + (2 \times 8) = 14$$

With binary activation function, the output of the neuron is
Y (threshold) = 1

McCulloch-Pitts Neuron

Architecture

- This is a simplified model of real neurons, known as a Threshold Logic Unit



- A set of input connections brings in activations from other neurons
- A processing unit sums the inputs, and then applies a non-linear activation function.
- The output line transmits the result to other neurons

Characteristics

- The activation of a McCulloch Pitts neuron is binary.
- Neurons are connected by directed weighted paths.
- A connection path is excitatory if the weight on the path is positive else its inhibitory.
- All excitatory connections to a neuron have the same weights.
- Each neuron has a fixed threshold:
  - f(n) = 1 if n >= θ
    0 if n < θ
- The threshold is set so that inhibition is absolute.

The McCulloch-Pits Neuron

- The equation for the output of a McCulloch-Pitts neuron as a function of 1 to n inputs is written as:

$$\text{Output} = \text{sgn}\left(\sum_{i=1}^{n} \text{Input}_i - \Phi \right)$$

- Where Φ is the neuron's activation threshold.

$$\text{If} \sum_{i=1}^{n} \text{Input}_i \geq \Phi \quad \text{then Output} = 1$$

$$\text{If} \sum_{i=1}^{n} \text{Input}_i < \Phi \quad \text{then Output} = 0$$

Examples

- Train a McCulloch-Pitts neural network to perform the OR function.
- Train a McCulloch-Pitts neural network to perform the AND function.
- Train a McCulloch-Pitts neural network to perform the AND NOT function.

Limitation of MP-neurons

- *Problems with MP-neurons*
- Weights and thresholds are analytically determined.
- Cannot learn
- Very difficult to minimize size of a network
- What about non-discrete and/or non-binary tasks?

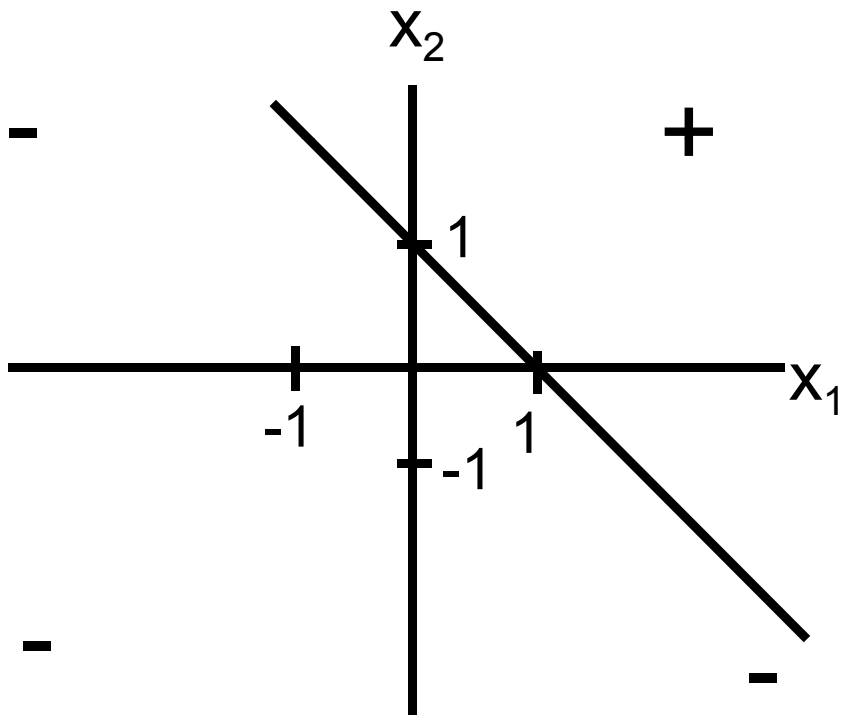
Solution is Perceptron Model.

Linear Separability

- A single layer neural network can only learn linear separable problems.
- Multilayer nets using a linear activation function have the same problem.
- In linear separable problems the region where y is positive, i.e. the neuron fires, is separated from the region where y is negative, i.e. where the neuron does not fire, by the line :

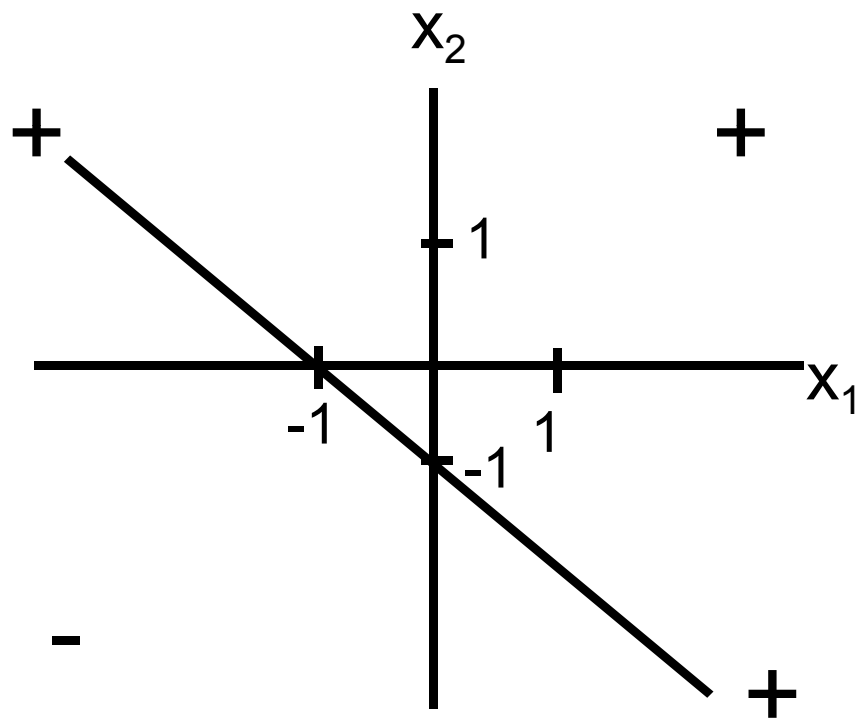
$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$

Graph for the AND Function



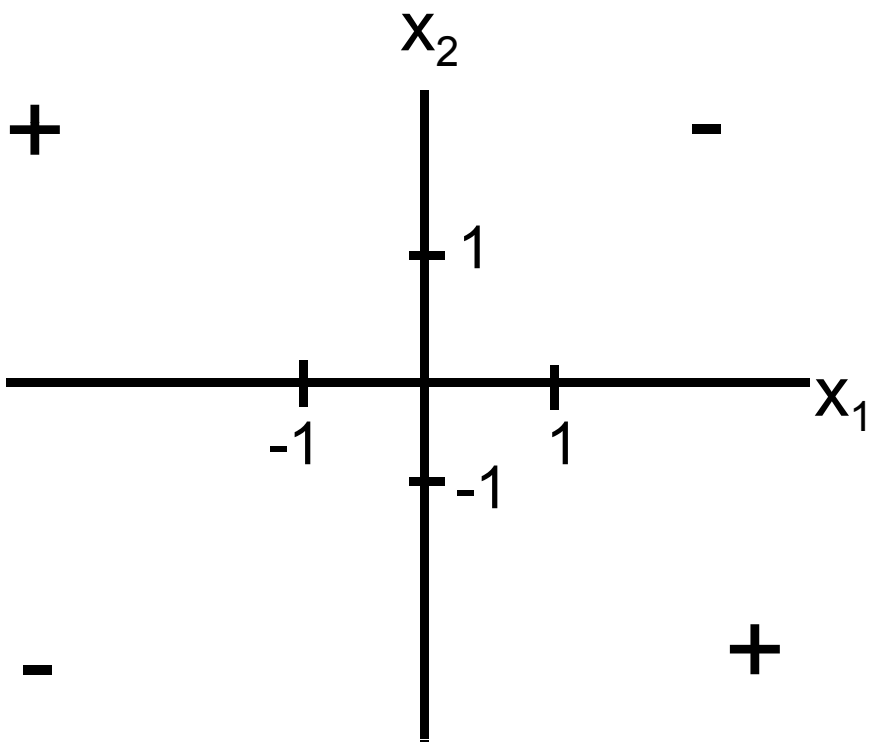
The AND function
is linearly separable

Graph for the OR Function



The OR function
is linearly separable

Graphic for the XOR Function



The XOR function
is not linearly separable

Hebbian Learning

- When axon of cell A is near enough to excite cell B and firing takes place repeatedly, then some metabolic or changes takes place in one or both cell , such that A's efficiency is increased.

- Hebb proposed a rule based on correlative weight adjustment.

- According to the Hebb rule, the weight vector is found to increase proportionately to the product of the input and learning signal(neuron's output).

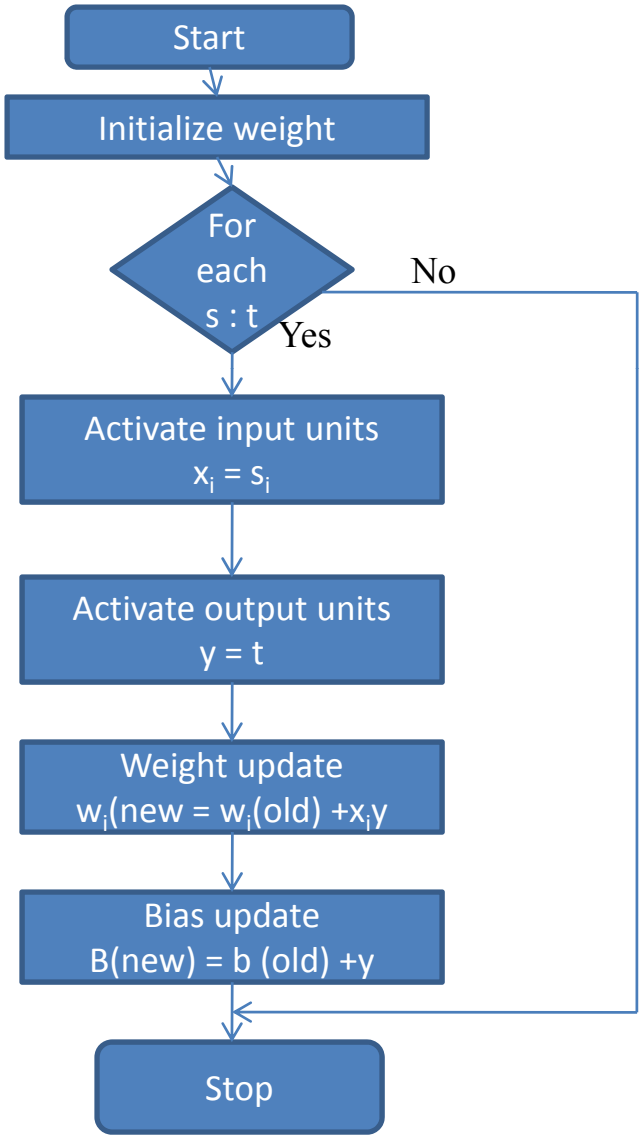
- In Hebb learning, if two interconnected neurons are 'on' simultaneously then the weight associated with these neurons can be increased by the modification made in their synaptic strength. The weight update is given by:

$$\text{i.e.} \qquad w_i(\text{new}) = w_i\,(\text{old}) + x_i y$$

- Hebb rule is more suited for bipolar data.

Flow chart of Hebb training Algo

- Let s:t refers to each training input and target output pair.



Popular neural systems

- ADALINE (Adaptive Linear Neural Element)
- ART (Adaptive Resonance Theory)
- AM (Associative Memory)
- BAM (Bidirectional Associative Memory)
- Boltzmann machines
- BSB (Brain – State-in-a-Box)
- Cauchy machines
- Hopfield Network
- LVQ (Learning Vector Quantization)
- Neoconition
- Perceptron
- RBF (Radial Basis Function)
- SOFM (Self –organizing Feature Map)