Lecture 8

Neural Network

→ Sub-branch of Deep learning, mimics human brain & neurons. Basic concepts cover differen fields such as CS, EE & Electronics.

→ NN/or ANN have further enhancements such as CNN, RNN, transformative learning.

→ Applications range from forcasting, Data minning & processing, traffic control, face recognition (image processing), control systems, signal processing & industrial automation.

Properties

→ Adaptive learning: NN have the ability to learn how to do tasks ~~based~~ based on given data/experience

→ Self Organization: NN can create their own organization or representation of data during training (eg. library).

→ Real Time Organization: NN have the ability to carry out computation in parallel in real time

→ Fault tolerance: If a particular or a group of neurons are damaged, some network capabilities are retained becz of its distributed architecture.

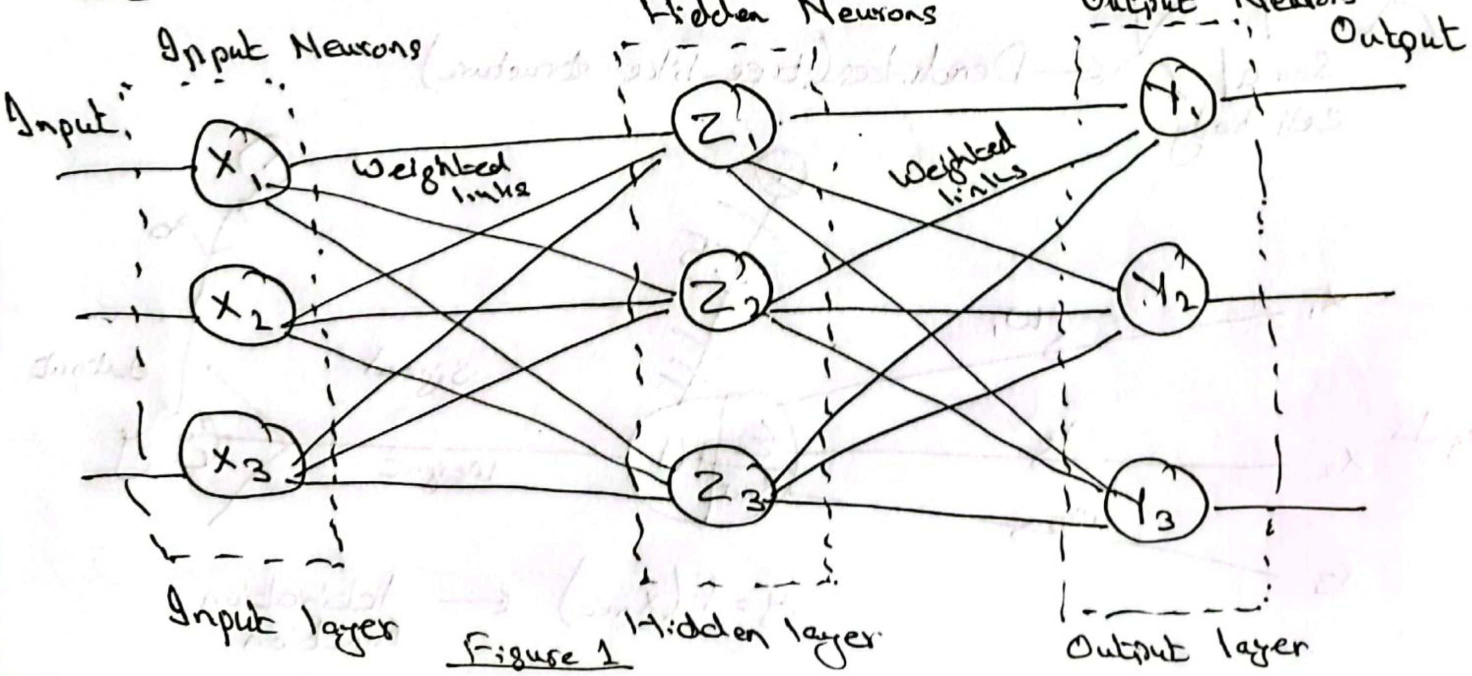
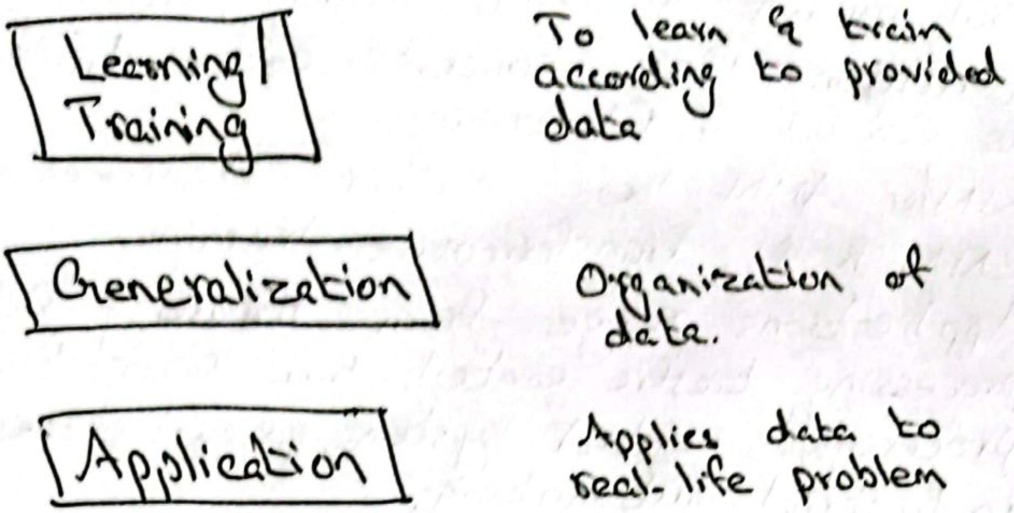


Figure 1

Objective

Parallel/distributed processing. All neurons are working.



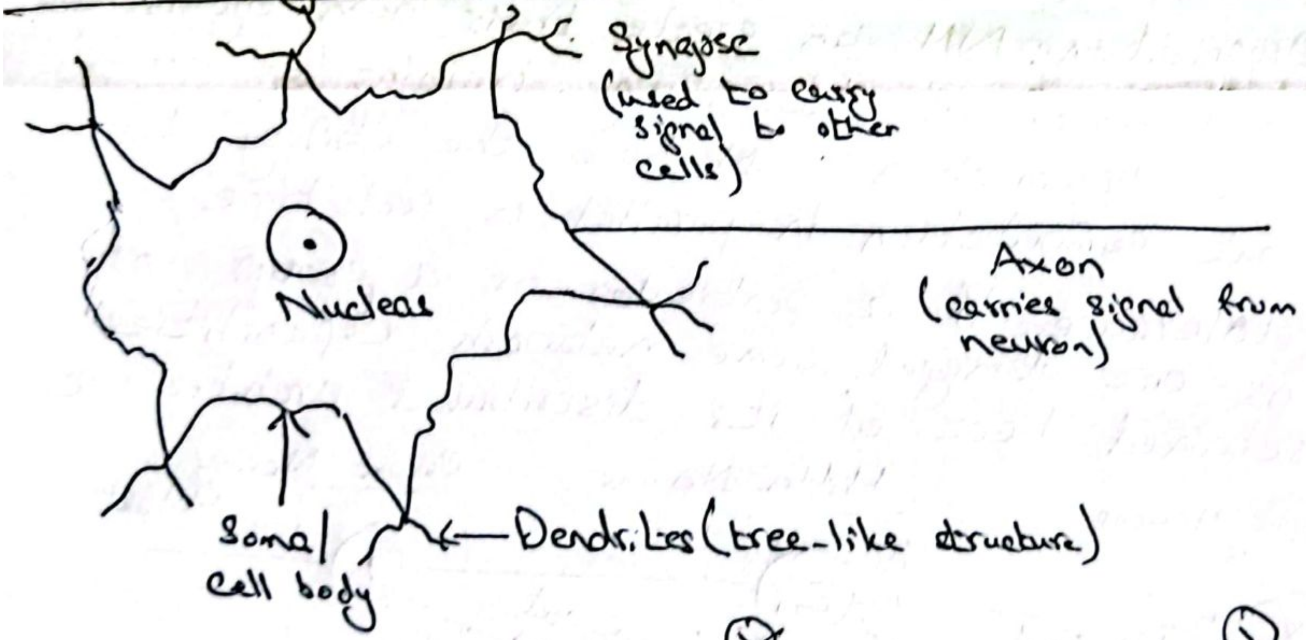
Learning/Training — To learn & train according to provided data

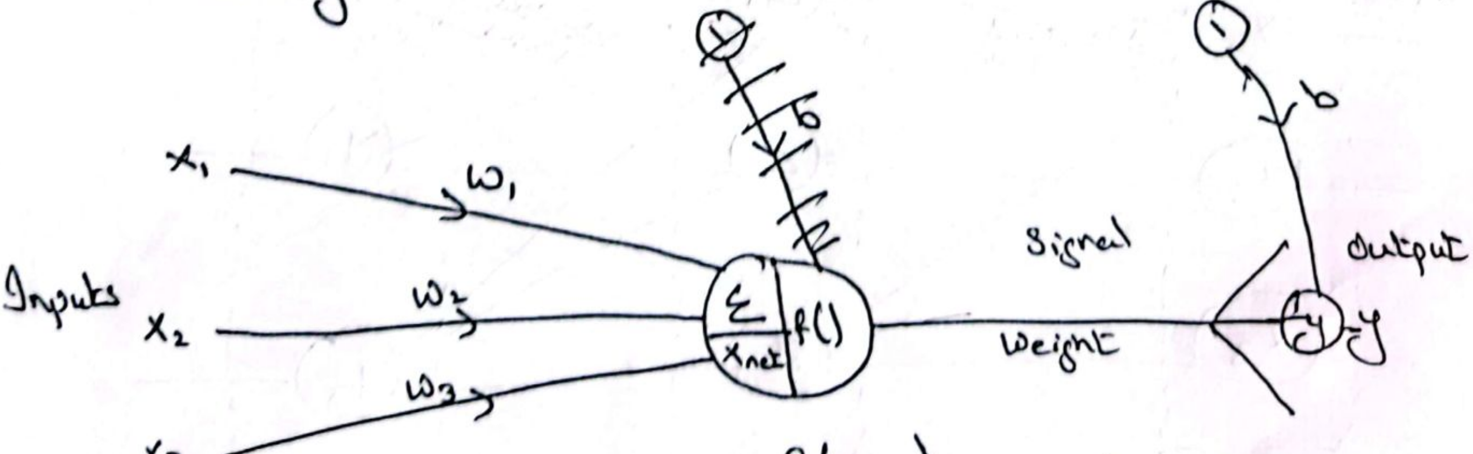
Generalization — Organization of data.

Application — Applies data to real-life problem

Figure 2

NN/Biological Neuron:


Synapse (used to carry signal to other cells)
Nucleas
Axon (carries signal from neuron)
Soma/cell body
Dendrites (tree-like structure)



$y = f(x_{net})$ ← Activation function

$$x_{(net)} = x_1 w_1 + x_2 w_2 + x_3 w_3 = \sum_{i=1}^{n} x_i w_i$$

$$y = x_1 w_1 + x_2 w_2 + x_3 w_3 + b.$$

Where
Cell = Neuron/Node
Dendrites/Synapse = Weights
Soma = Net Input
Axo = Output.

-> Weight is added to alter the input. Input signal is multiplied with weight.

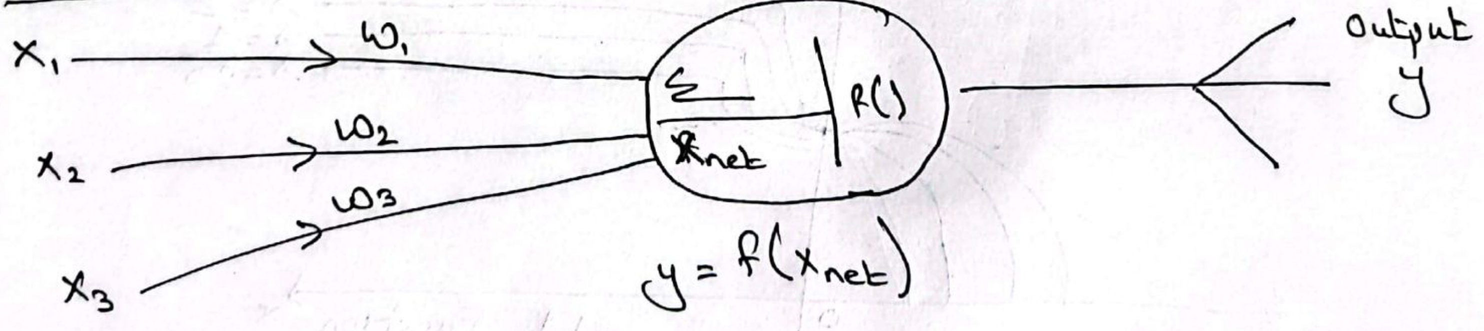
-> Bias: is a constant signal value which is added, and is ~~ke~~ like another weighted link with constant value.

-> Activation Function: is associated with each neuron & determines the input-output relationship for that neuron. It can be either linear/non-linear.

-> Threshold: is a predefined set or constant value depending on which output of NN is determined.

$$f(x) = \begin{cases} -1 & x < \theta \\ 1 & x \geq \theta \end{cases}$$ where θ is fixed Threshold value.

Activation Function:



-> Each neuron has certain AF associated with it.

-> The AF is applied over the net input to the node.

-> There are diff types of AF (linear/non-linear) and is selected on the type of output desired.

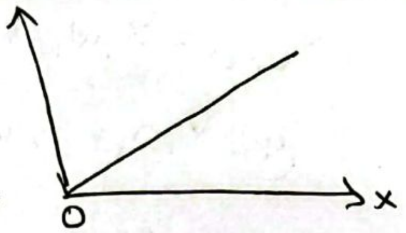
-> Common types of AF are:

(e)

1) Identity function:

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

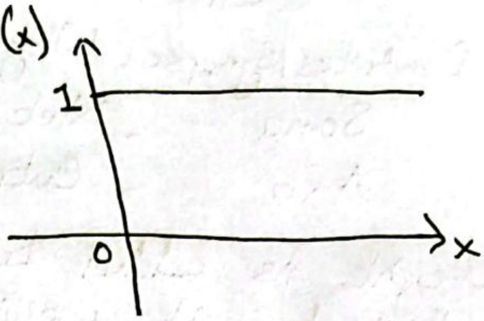
The ~~input~~ output is same as output.
Mostly used in input layer.



2) Binary Step Function:

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

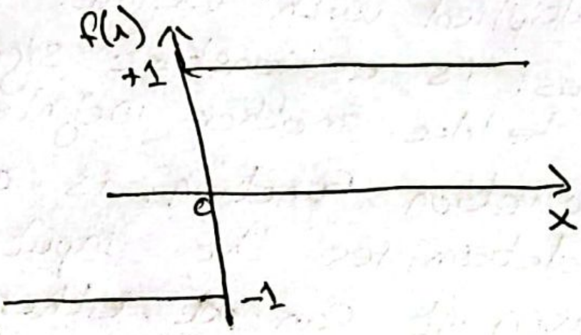
where θ = Threshold value



3) Bipolar Step Function

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

where θ = Threshold value.
Mostly used in digital systems.



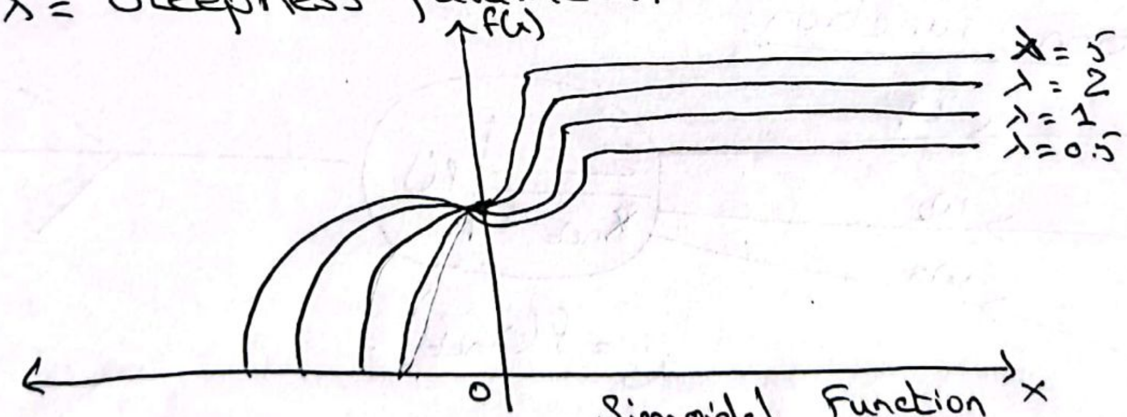
4) Binary Sigmoidal Function

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

5) Bipolar Sigmoidal Function

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

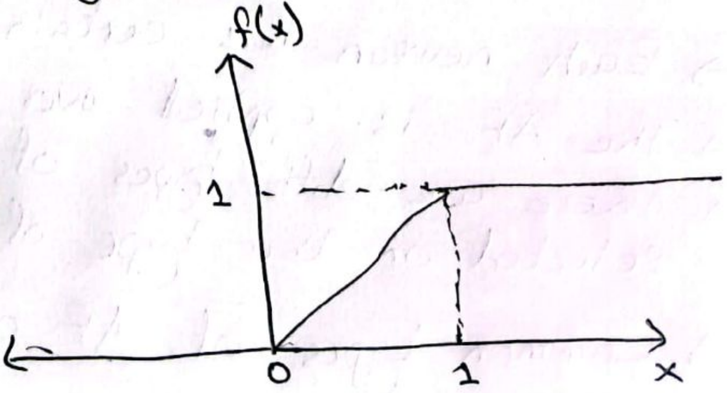
λ = Steepness parameter.



Sigmoidal Function

6) Ramp Function

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



Types of Learning:

Learning/ Training

Generalization

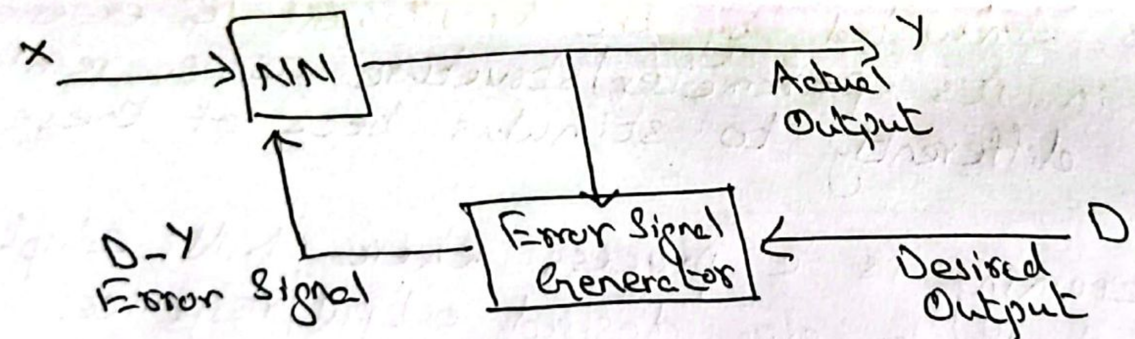
Application

→ A process where an NN adept & adjusts itself to any stimulus to it by making structural or parametric (weight, bias, network architecture) adjustments in order to generate the desired output.

→ ~~Two k~~ Three types of learning processes:

Supervised Learning:

→ Actual involvement of a supervisor.



Unsupervised & Reinforcement:

→ Unsup learning where NN makes mistakes & learn from those

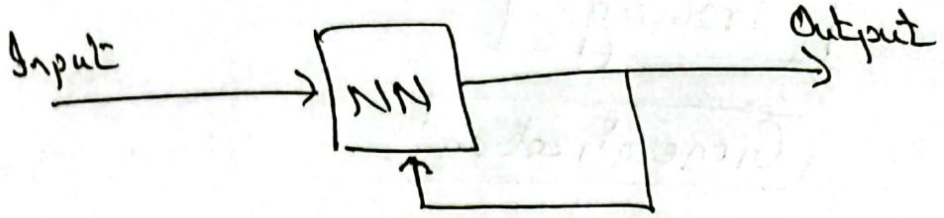
→ It combines all similar types/nature of data are combined together to form groups/clusters of data.

→ When new input is applied, the NN compares it to all existing groups/clusters to check if it belongs to any existing one and stores it in it.

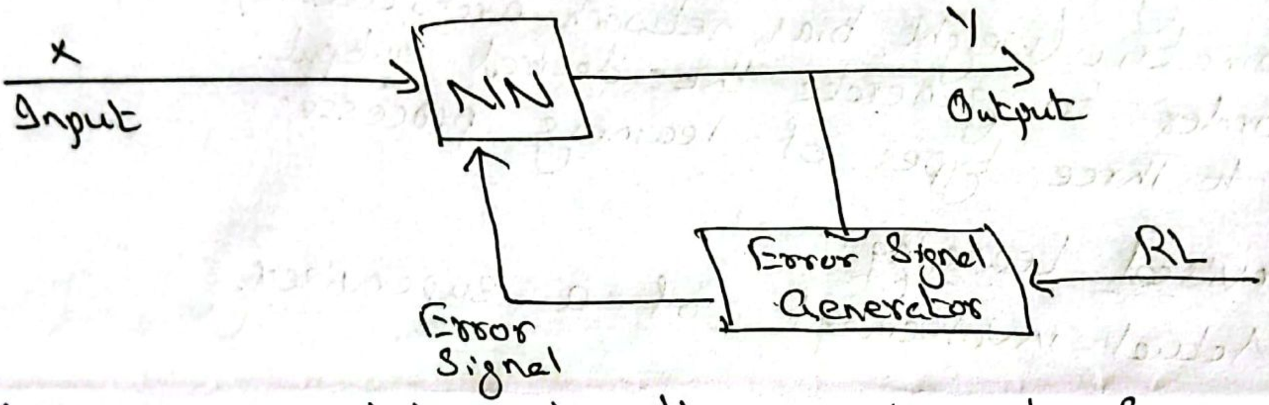
→ If input data is of new/different nature, the NN creates a new group to share it in it.

→ The NN itself discover pattrens, establishes relationships

betw data types & simultaneously undergoes changes
in its parameters.



-> For RL, is same as supervised learning. The diff is that feedback in sup Learning is in absolute terms. In RL, the feedback is ~~in~~ relative/probablistic terms.



-> NN is simulated by the enviroment & experience changes in it's parameter/structure as a result

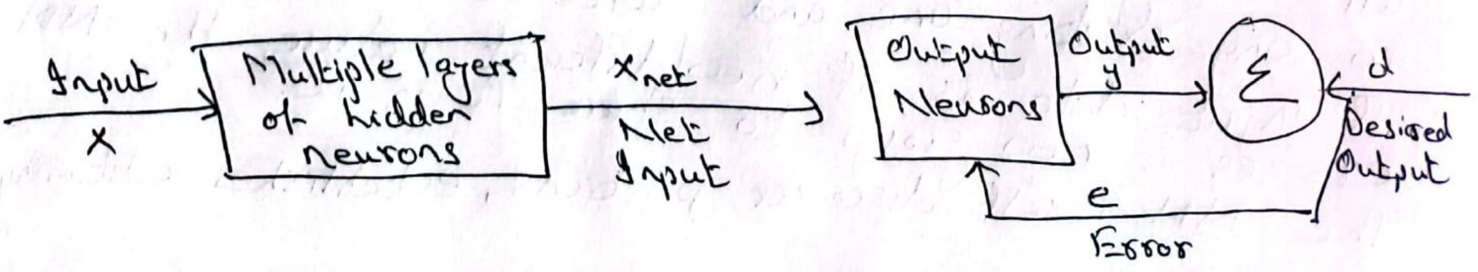
-> Responds differently to stimulus becz of these changes.

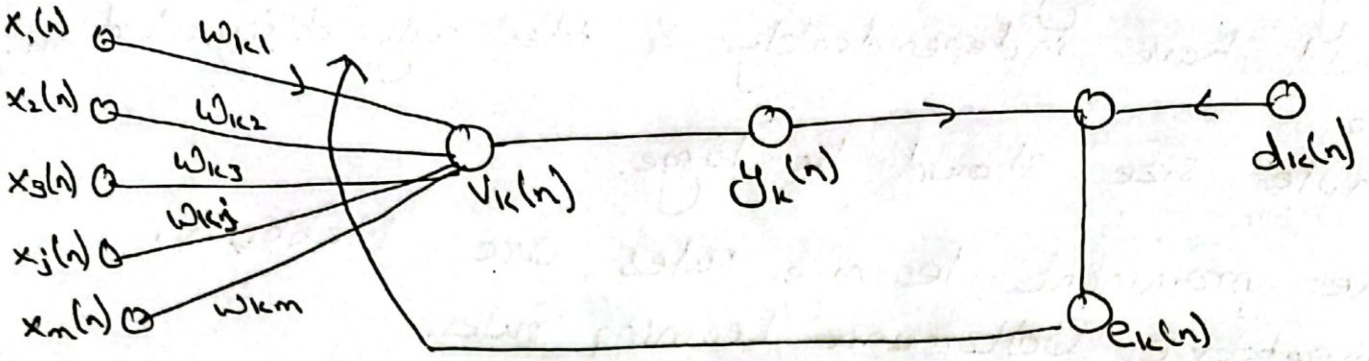
-> Learning/training is a process where NN adapts & adjusts itself to give desired output/responce

1) Error Correction Learning

consits of 1). Input and output layers of neurons.
2). Input Signal
3). Comparator
4). Devised or target output

-> It is a parmeter based learning.





Depending on the output & desired or target input the error signal is given by

$$e_k(n) = d_k(n) - y_k(n)$$

Based on "error signal", the change in synaptic weight is given by (known as delta or window-hoff rule)

$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n)$$

where η = learning rate
x_j = Input signal
e_k = Error Signal

Memory based rules

→ Concepts or experiences are classified by similarity with previously seen concept/experiences.

→ Strong training data items (clustering, classify, pattrens etc)

→ Generally based on "nearest neighbour rule", but also "space decomposition" & clustering techniques.

→ The training pattren is an input vector 'x' with components $\{x_1, x_2, ----, x_n\}$ where desired outputs 'd' is $\{d_1, d_2, -----, d_n\}$

→ It is stored in a large memory in the form of correctly classified input-output examples.

$$\{(x_i, d_i)\}_{i=1}^{N}$$

meaning we have

$$x_1 \rightarrow d_1$$
$$x_2 \rightarrow d_2$$
$$x_3 \rightarrow d_3$$

→ Finding similarity of test(new) date (through Ed)

→ Should have independently & identically distributed data

→ Sample size should be large.

→ Other prominent learning rules are Hebbian, Competitive, Boltzmann Learning rules.