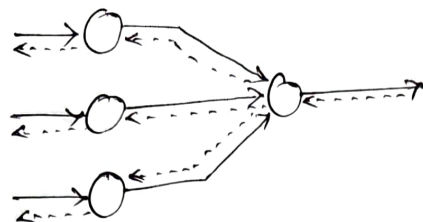
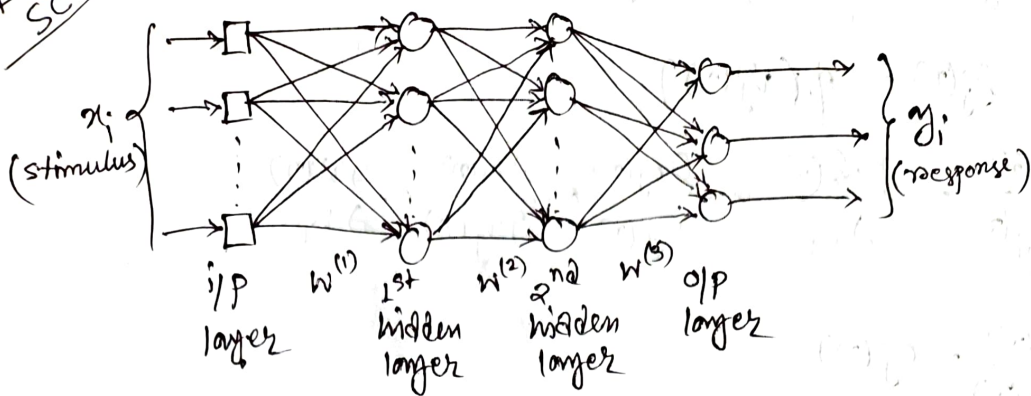


08/10/25
SC

Multi-layer perceptron:



Two basic signal flows in MLP

Back-propagation Algo:

j th neuron \rightarrow output

$$e_j(m) = d_j(m) - y_j(m)$$

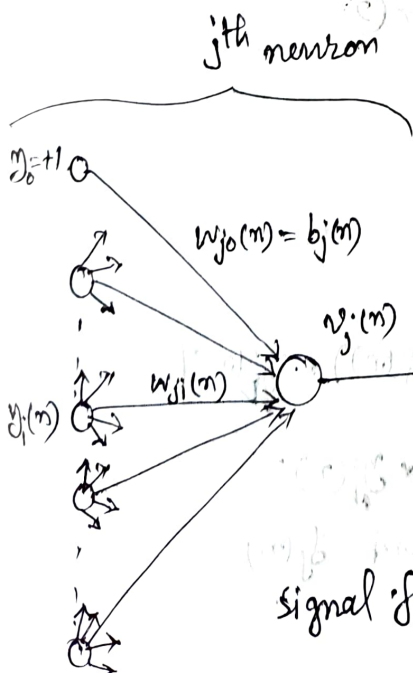
$$E(m) = \frac{1}{2} \sum_{j \in c} e_j^2(m)$$

class

$y_j(m) \rightarrow$ o/p of j th neuron

$d_j(m) \rightarrow$ desired value of j th neuron

$$E_{avg} = \frac{1}{N} \sum_{n=1}^N E(m)$$



signal flow graph when j is off to neuron

$$v_j(n) = \sum_{i=1}^n w_{ji}(n) y_i(n)$$

$$y_j(n) = \phi(v_j(n))$$

$$\frac{\partial \epsilon(n)}{\partial w_{ji}(n)} = \frac{\partial \epsilon(n)}{\partial e_j(n)} * \frac{\partial e_j(n)}{\partial y_j(n)} * \frac{\partial y_j(n)}{\partial v_j(n)} * \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

$$\frac{\partial \epsilon(n)}{\partial e_j(n)} = e_j(n)$$

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1$$

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \phi'(v_j(n))$$

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n)$$

$$\therefore \frac{\partial \epsilon(n)}{\partial w_{ji}(n)} = e_j(n) * (-1) * \phi'(v_j(n)) * y_i(n) \longrightarrow \textcircled{1}$$

According to delta learning rule (i.e. steepest descent algo.)

$$\Delta \vec{w}(n) = -\eta \vec{g}(n)$$

$$= -\eta \nabla \epsilon(n)$$

$$\Delta w_{ij}(n) = -\eta \frac{\partial \epsilon(n)}{\partial w_{ji}(n)} \longrightarrow \textcircled{2}$$

put $\textcircled{1}$ into $\textcircled{2}$

$$\Delta w_{ji}(n) = -\eta \frac{\partial \epsilon(n)}{\partial w_{ji}(n)}$$

$$= -\eta [-e_j(n) * \phi'(v_j(n)) * y_i(n)]$$

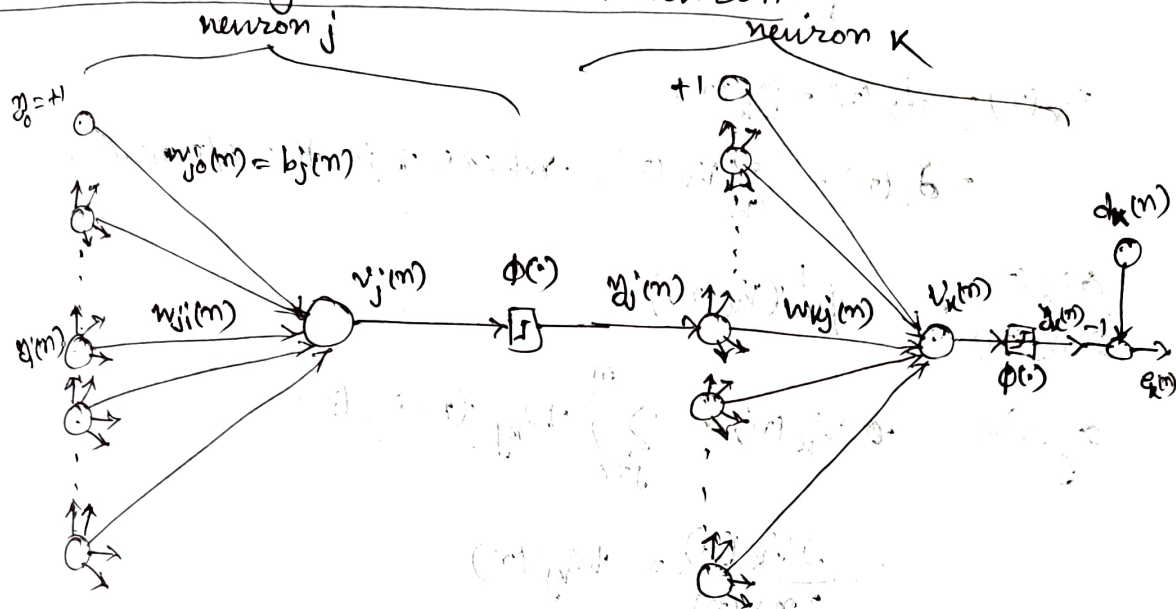
$$= \eta \underbrace{e_j(n) * \phi'(v_j(n)) * y_i(n)}_{\text{local gradient } \delta_j(n)}$$

local gradient $\delta_j(n)$

$$\begin{aligned}
 \delta_j(n) &= e_j(n) \phi'(v_j(n)) = -e_j(n) * (-1) * \phi'(v_j(n)) \\
 &= - \frac{\partial e(n)}{\partial e_j(n)} \cdot \frac{\partial e_j(n)}{\partial y_j(n)} \cdot \frac{\partial y_j(n)}{\partial v_j(n)} \\
 &= - \frac{\partial e(n)}{\partial v_j(n)}
 \end{aligned}$$

$$\Delta w_{ji}(n) = \eta \cdot \delta_j(n) \cdot y_i(n) \quad \text{when } j \text{ is o/p node}$$

when neuron j is a hidden neuron.



$$\begin{aligned}
 \delta_j(n) &= - \frac{\partial e(n)}{\partial y_j(n)} * \frac{\partial y_j(n)}{\partial v_j(n)} \\
 &= - \frac{\partial e(n)}{\partial y_j(n)} \cdot \phi'(v_j(n))
 \end{aligned}$$

09/10/23
soft computing

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n)$$

where,

$$\delta_j(n) = - \frac{\partial \mathcal{E}(n)}{\partial y_j(n)} * \phi'(v_j(n))$$

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = \sum_k e_k \cdot \frac{\partial e_k(n)}{\partial y_j(n)}$$

$$= \sum_k e_k(n) \frac{\partial e_k(n)}{\partial v_k(n)} * \frac{\partial v_k(n)}{\partial y_j(n)}$$

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

$$= d_k(n) - \phi(v_k(n)) \text{ where } k \text{ is the o/p neuron}$$

$$\frac{\partial e_k(n)}{\partial v_k(n)} = - \phi'_k(v_k(n))$$

$$\text{again, } v_k(n) = \sum_{j=0}^m w_{kj}(n) y_j(n)$$

$$\therefore \frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n)$$

$$\text{then, } \frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = - \sum_k e_k(n) \cdot \phi'(v_k(n)) w_{kj}(n)$$

$$= - \sum_k \delta_k(n) w_{kj}(n)$$

$$\delta_j(n) = \phi'(v_j(n)) \sum_k \delta_k(n) w_{kj}(n)$$

$$\boxed{\delta_j(n) = \phi'(v_j(n)) \sum_k \delta_k(n) w_{kj}(n)}$$

where j is
hidden neuron

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n)$$

- ① Forward pass
- ② Backward pass

① Forward pass:

$$v_j(n) = \sum_{i=0}^m w_{ji}(n) y_i(n) \text{ where, } y_i(n) = x_i(n)$$

output,

$$y_j(n) = \phi(v_j(n))$$

jth neuron output $y_j(n) = y_j(n)$

② Backward pass:

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}_{avg}}{\partial w_{ji}}$$

$$w_{ji}(n+1) = w_{ji}(n) + \Delta w_{ji}(n)$$

Activation Fn.

$$\Delta w_{ji}(n) = \eta \cdot \underset{\substack{\uparrow \\ \text{local gradient}}}{y_j(n)} \cdot \underset{\substack{\nwarrow \text{ i/p } \\ \text{local gradient}}}{y_i(n)}$$

(i) Logistic function

(ii) Hyperbolic tangent function

(i) Logistic Fn:

$$\phi_j(v_j(n)) = \frac{1}{1 + e^{-a v_j(n)}} \text{ where } a > 0$$

$$\phi_j'(v_j(n)) = \frac{a \cdot e^{-a v_j(n)}}{[1 + e^{-a v_j(n)}]^2}$$

$$\phi_j'(v_j(n)) = a y_j(n) [1 - y_j(n)]$$

Neuron j is o/p node:

$$\begin{aligned}d_j(n) &= e_j(n) \cdot \phi'(v_j(n)) \\&= [d_j(n) - o_j(n)] \cdot \alpha \cdot o_j(n) [1 - o_j(n)] \\&\quad \text{when } y_j(n) = o_j(n) \\&= \alpha [d_j(n) - o_j(n)] o_j(n) [1 - o_j(n)]\end{aligned}$$

Neuron j is hidden node:

$$\begin{aligned}d_j(n) &= \phi'(v_j(n)) \cdot \sum_k d_k(n) w_{kj}(n) \\&= \alpha y_j(n) [1 - y_j(n)] \sum_k d_k(n) w_{kj}(n)\end{aligned}$$

(ii) Hyperbolic Tangent fn.:

$$\phi_j(v_j(n)) = a \tanh(\beta v_j(n)), \quad (a, \beta) > 0$$

$$\begin{aligned}\phi'_j(v_j(n)) &= \alpha \beta \operatorname{sech}^2(\beta v_j(n)) \quad [\operatorname{sech}^2(x) : \text{hyperbolic secant squared}] \\&= \alpha \beta \cdot (1 - \tanh^2(\beta v_j(n))) \\&= \frac{\beta}{a} [a - y_j(n)] [a + y_j(n)]\end{aligned}$$

→ If j is output node, $y_j(n) = o_j(n)$

$$\begin{aligned}d_j(n) &= e_j(n) \cdot \phi'(v_j(n)) \\&= \frac{\beta}{a} [d_j(n) - o_j(n)] [a - o_j(n)] [a + o_j(n)]\end{aligned}$$

If j is hidden node:

$$\begin{aligned}d_j(n) &= \phi'(v_j(n)) \cdot \sum_k d_k(n) w_{kj}(n) \\&= \frac{\beta}{a} [a - y_j(n)] [a + y_j(n)] \sum_k d_k(n) w_{kj}(n)\end{aligned}$$

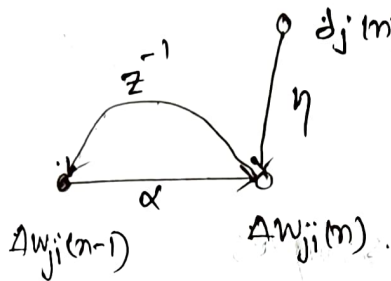
Rate of Learning:

Include a momentum term α

after including momentum term α

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta d_j(n) y_i(n)$$

where, momentum term α is positive number,
is also called momentum constant



Signal flow graph

Modes of Training:

(i) Sequential Mode: Data are feed to the neuron sequentially

$$\{(x(1), d(1)), (x(2), d(2)), \dots, (x(N), d(N))\}$$

(ii) Batch Mode:

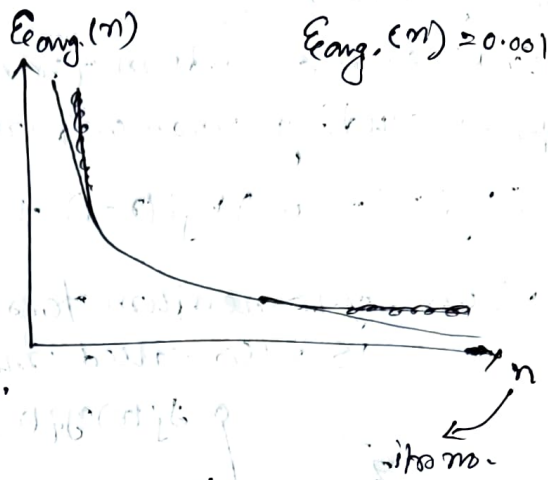
$$E_{avg} = \frac{1}{2N} \cdot \sum_{n=1}^N \sum_{j \in C} e_j^2(n)$$

$$\Delta w_{ji}(n) = -\eta \cdot \frac{\partial E_{avg}}{\partial w_{ji}(n)}$$

$$= -\frac{\eta}{N} \cdot \sum_{n=1}^N e_j(n) \cdot \frac{\partial e_j(n)}{\partial w_{ji}(n)}$$

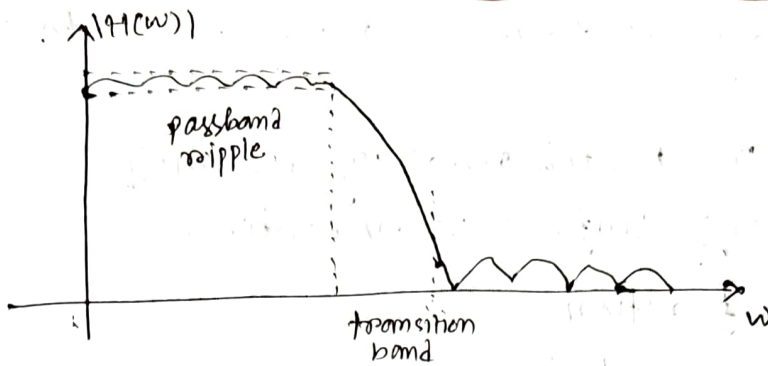
Stopping Criteria:

- (i) Gradient vector is sufficiently small
- (ii) when absolute rate of change of the avg. square errors per epoch is sufficiently small.



Summary:

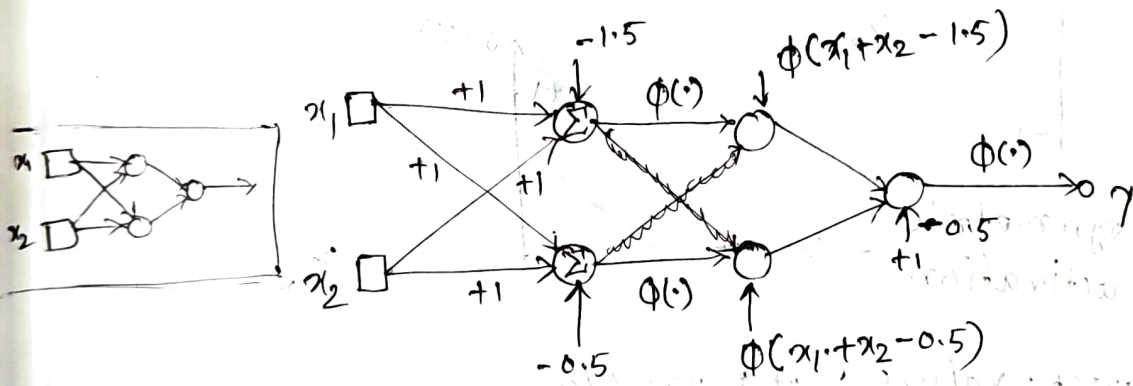
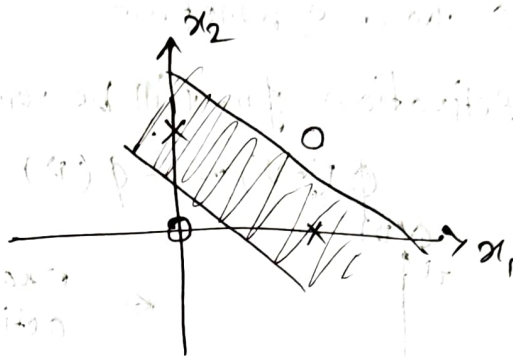
- i) Initialization
- ii) presentation of training samples
- iii) forward computation
- iv) Backward
- v) Iteration Interaction



10/10/25
Soft computing

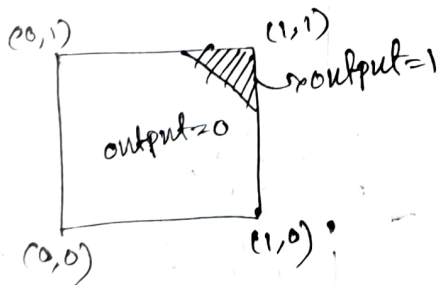
XOR problem:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

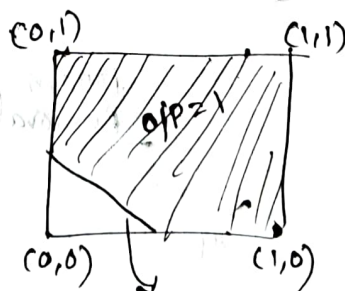


Assume $\phi(\cdot)$ hard-limit

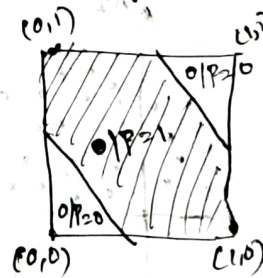
Decision Boundary:



@ neuron-1



@ neuron-2



@ neuron-3

Heuristics to move to BP-MLP algo. with better performance.

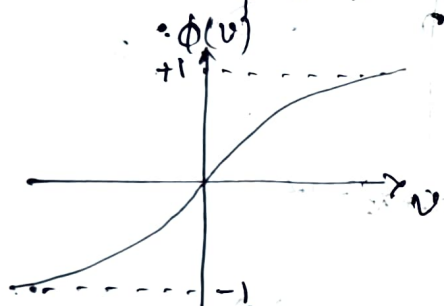
- (i) Sequential vs Batch mode of learning
- (ii) Maximizing Information content
- (iii) Activation function

Ⓐ Anti-symmetric

Ⓑ Non-symmetric

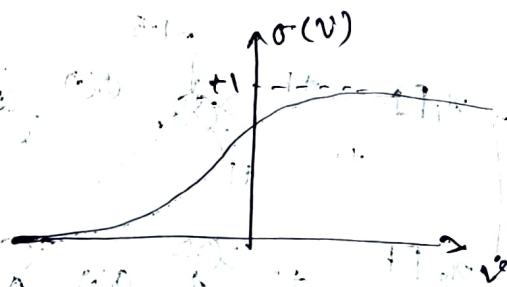
Activation fn. will be anti-symmetric if

$$\phi(-v) = -\phi(v)$$



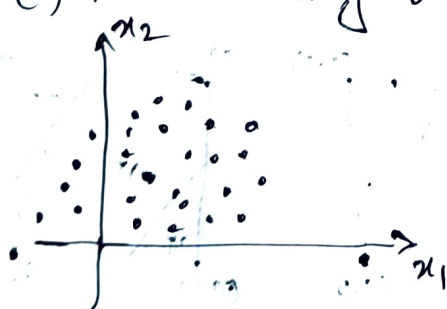
example of anti-symmetric activation fn. $\phi(v) = a \tanh(bv)$

Non-symmetric activation

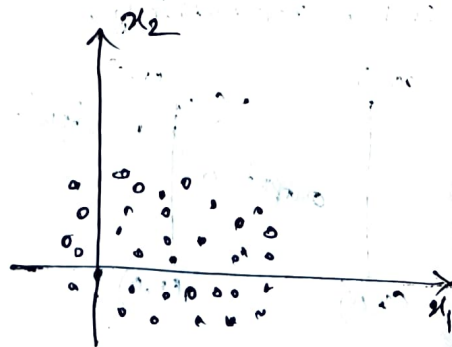


(iv) Target values: ± 1 or $1/0$

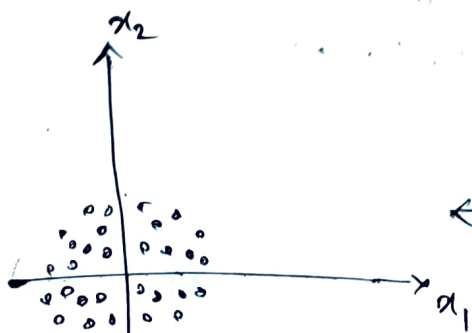
(v) Normalizing the inputs



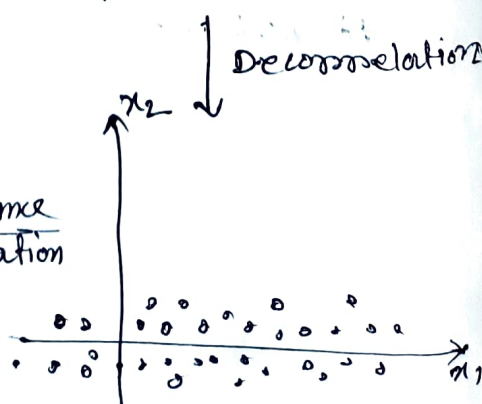
Mean Removal



Decorrelation



Covariance equalization



⑥ Initialization of weights:

Assume, $b=0$

$$v_j = \sum_{i=1}^m w_{ji} y_i$$

i/p

→ Assume, i/p applied with zero mean & unit variance

→ Assume, i/p's are uncorrelated

$$\mu_y = E[y_i] = 0 \quad \forall i$$

$$\begin{aligned} \sigma_y^2 &= E[(y_i - \mu_y)^2] \\ &= E[y_i^2] = 1 \quad \forall i \quad [\because \mu_y = 0] \end{aligned}$$

$$E[y_i y_k] = \begin{cases} 1 & \text{for } k=i \\ 0 & \text{for } k \neq i \end{cases}$$

$\mu_w = E[w_{ji}] = 0 \quad \forall (j,i) \text{ pairs}$
and variance

$$\begin{aligned} \sigma_w^2 &= E[(w_{ji} - \mu_w)^2] \\ &= E[w_{ji}^2] \quad \forall (j,i) \text{ pairs} \end{aligned}$$

Synaptic weights are drawn from uniformly distributed set of numbers with zero mean.

$\mu_w = E[w_{ji}] = 0$
 $\forall (j,i) \text{ pairs}$

Mean & variance of v_j

$$\mu_v = E[v_j] = E\left[\sum_{i=1}^m w_{ji} y_i\right] = \sum_{i=1}^m E[w_{ji}] E[y_i] = 0$$

$$\sigma_v^2 = E[(v_j - \mu_v)^2] = E[v_j^2]$$

$$= E\left[\sum_{i=1}^m \sum_{k=1}^m w_{ji} y_i w_{jk} y_k\right]$$

$$= \sum_{i=1}^m \sum_{k=1}^m E[w_{ji} w_{jk}] E[y_i y_k]$$

$$= \sum_{i=1}^m E[w_{ji}^2] \quad \text{for } i=k$$

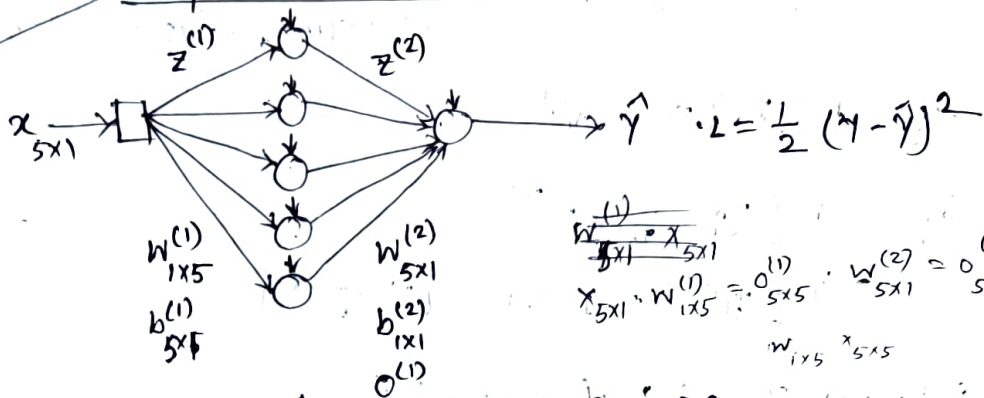
$= m \sigma_w^2$ where m : # synaptic connection of neuron

$$\sigma_w^2 = \frac{1}{m} \sigma_v^2 \Rightarrow \sigma_w = \frac{1}{\sqrt{m}} \sigma_v \quad \text{for } \sigma_v = 1, \sigma_w = m^{-1/2}$$

- (vi) learning from hints
- (vii) learning rate

13/10/25
Soft computing
Lab

Experiment - 06



$$W^{(1)} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} & w_{15} \end{bmatrix}_{1 \times 5}$$

$$x_{5 \times 1} \cdot W^{(1)}_{1 \times 5} = o^{(1)}_{5 \times 5}$$

$$W^{(2)} = \begin{bmatrix} w_{21} & w_{22} & w_{23} & w_{24} & w_{25} \end{bmatrix}_{5 \times 1}$$

$$o^{(1)}_{5 \times 5} \cdot W^{(2)}_{5 \times 1} = z^{(2)}_{5 \times 1}$$

$$\frac{\partial L}{\partial W^{(2)}} = \frac{\partial L}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_2} * \frac{\partial z_2}{\partial W^{(2)}}$$

$$= -(y - \hat{y}) * \sigma'(z_2) * o_1$$

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_2} * \frac{\partial z_2}{\partial o_1} * \frac{\partial o_1}{\partial z_1} * \frac{\partial z_1}{\partial W_1}$$

$$= -(y - \hat{y}) * \sigma'(z_2) * W_2 * \sigma'(z_1) * x$$

$$\sigma'(z_2) = \sigma(z_2) [1 - \sigma(z_2)]$$

$$\frac{\partial L}{\partial \hat{y}} = 1 - (y - \hat{y})$$

$$\frac{\partial \hat{y}}{\partial z_2} = \sigma'(z_2)$$

$$\frac{\partial z_2}{\partial W_2} = o_1$$

$$\frac{\partial z_2}{\partial o_1} = W_2$$

$$\frac{\partial o_1}{\partial z_1} = \sigma'(z_1)$$

$$\frac{\partial z_1}{\partial W_1} = x$$

$$z_1 = x \cdot W_1$$

$$x_{5 \times 1} \Rightarrow x'_{5 \times 2} \cdot w^{(1)}_{2 \times 5} = -w^{(1)T}_{5 \times 2} \cdot x'^T_{2 \times 5} = z^{(1)}_{5 \times 5}$$

$$o^{(1)}_{5 \times 5} \Rightarrow o^{(1)1}_{5 \times 6} \cdot w^{(2)}_{6 \times 1} \Rightarrow z^{(2)}_{5 \times 1}$$

$$(y - o_2)_{5 \times 1} \cdot o^{(2)}_{5 \times 1} = \text{o/p gradient}$$

$$\Delta w_{5 \times 1} \cdot W^{(2)}_{5 \times 1} = \begin{bmatrix} \Delta w_{11} & \Delta w_{12} & \Delta w_{13} & \Delta w_{14} & \Delta w_{15} \end{bmatrix}_{5 \times 1} \cdot \begin{bmatrix} w_{21} & w_{22} & w_{23} & w_{24} & w_{25} \end{bmatrix}_{5 \times 1}$$

$$\begin{bmatrix} \Delta w_{11} & \Delta w_{12} & \Delta w_{13} & \Delta w_{14} & \Delta w_{15} \end{bmatrix}_{1 \times 5} \cdot \begin{bmatrix} w_{21} & w_{22} & w_{23} & w_{24} & w_{25} \end{bmatrix}_{5 \times 1} = \begin{bmatrix} \Delta w_{11} & \Delta w_{12} & \Delta w_{13} & \Delta w_{14} & \Delta w_{15} \end{bmatrix}_{1 \times 5}$$

$$\Delta w_{1 \times 5} \cdot W^{(2)}_{5 \times 1} = \begin{bmatrix} \Delta w_{11} & \Delta w_{12} & \Delta w_{13} & \Delta w_{14} & \Delta w_{15} \end{bmatrix}_{1 \times 5} \cdot \begin{bmatrix} w_{21} & w_{22} & w_{23} & w_{24} & w_{25} \end{bmatrix}_{5 \times 1} = \begin{bmatrix} \Delta w_{11} & \Delta w_{12} & \Delta w_{13} & \Delta w_{14} & \Delta w_{15} \end{bmatrix}_{1 \times 5}$$

$$1 \times 5 \cdot \begin{bmatrix} 5 \times 5 \end{bmatrix} \cdot \begin{bmatrix} 5 \times 1 \end{bmatrix} = 5 \times 5$$

15/10/25
soft computing

Radial Basis Function Network

● Covers theorem: A complex pattern classification problem in a high dimensional space ~~dimensionality~~ non-linearly is more likely to be linearly separable than in a low dimensional space.

$$\vec{x} = [x_1, x_2, \dots, x_N]$$

$\phi(x) \rightarrow$ hidden fn.

$$\begin{aligned} \Phi(x) &= [\phi_1(x), \phi_2(x), \dots, \phi_m(x)]^T \\ &= \{\phi_i(x)\}_{i=1}^m \rightarrow \text{feature space.} \end{aligned}$$

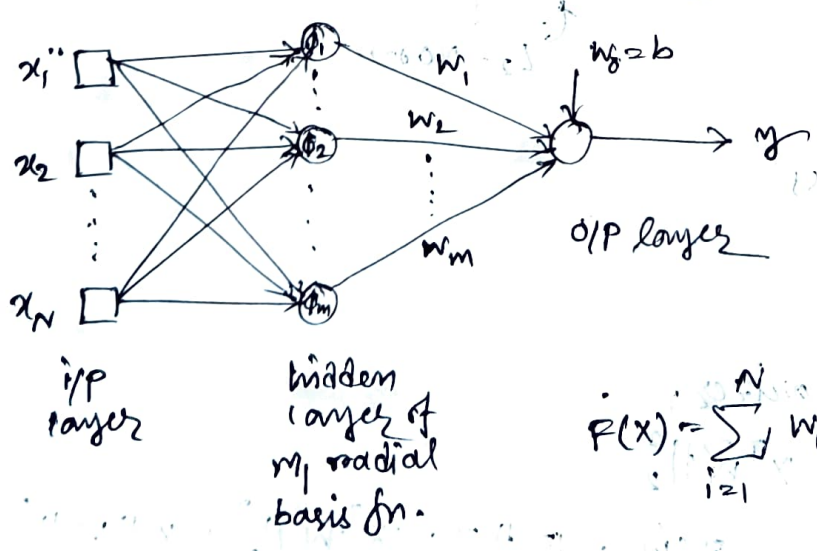
$$W^T \Phi(x) > 0 \quad ; \quad x \in C_1$$

$$W^T \Phi(x) < 0 \quad ; \quad x \in C_2$$

Separation hyperplane,

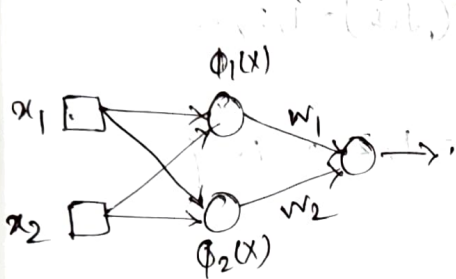
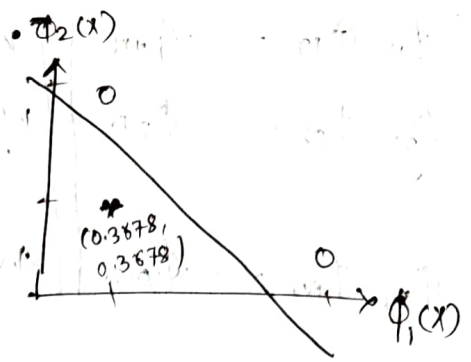
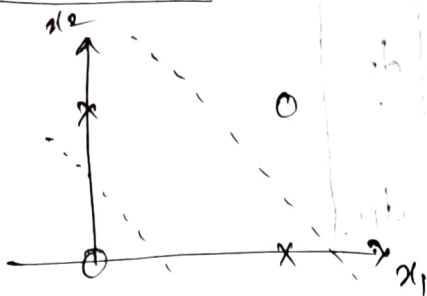
$$W^T \Phi(x) = 0$$

\hookrightarrow separating surface



$$F(x) = \sum_{i=1}^N w_i \phi(\|x - x_i\|)$$

XOR problem:



$x_2(x_1, x_2)$	γ	$\phi_1(x)$	$\phi_2(x)$
$(1, 1)$	0	1	0.1353
$(0, 1)$	1	0.3678	0.3678
$(0, 0)$	0	0.1353	1
$(1, 0)$	1	0.3678	0.3678

Gaussian fn.

$$\left. \begin{aligned} \phi_1(x) &= e^{-\|x - t_1\|^2} \\ \phi_2(x) &= e^{-\|x - t_2\|^2} \end{aligned} \right\} \begin{aligned} t_1 &= [1, 1]^T \\ t_2 &= [0, 0]^T \end{aligned} \quad \text{centres}$$

Interpolation problem:

$$S: \mathbb{R}^{m_0} \rightarrow \mathbb{R}^1$$

↑
mapping

Multivariable nonlinear mapping
Multivariable interpolation
a set of N different points.

$$\begin{aligned} \{x_i \in \mathbb{R}^{m_0}\}_{i=1,2,\dots,N} \\ \downarrow \\ \{d_i \in \mathbb{R}^1\}_{i=1,2,\dots,N} \end{aligned}$$

$$F(x_i) = d_i, \quad i=1,2,\dots,N$$

$$F(x) = \sum_{i=1}^N w_i \phi(\|x - x_i\|)$$

$$\{\phi(\|x - x_i\|)\}_{i=1,2,\dots,N}$$

↑
Radial basis function

$$\begin{bmatrix} \phi_{11} & \phi_{12} & \dots & \phi_{1N} \\ \phi_{21} & \phi_{22} & \dots & \phi_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{N1} & \phi_{N2} & \dots & \phi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix}$$

where $\phi_{ji} = \phi(\|x_j - x_i\|)$ $(j, i) = 1, 2, \dots, N$

$$\vec{\phi} = \{ \phi_{(j,i)} \mid (j,i) = 1, 2, \dots, N \}$$

$$\vec{\phi} \vec{w} = \vec{d}$$

$$\vec{w} = \vec{\phi}^{-1} \vec{d}$$

Micchelli's Theorem:

i) Multi-quadrics:

$$\phi(r) = \sqrt{r^2 + c^2} \quad \text{for some } c > 0 \quad \& \quad r \in \mathbb{R}$$

ii) Inverse multiquadrics:

$$\phi(r) = \frac{1}{\sqrt{r^2 + c^2}} \quad \text{for some } c > 0 \quad \& \quad r \in \mathbb{R}$$

iii) Gaussian form:

$$\phi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \quad \text{for some } \sigma > 0 \quad \& \quad r \in \mathbb{R}$$