CLASS | SUBJECT
ROLL NO. | DATE

Unit - 4

Introduction to machine learning

Part of AI

Training

* Task to learn to classify inputs according to a finite set of classifications.
* Learn from training data.
* Classify new data that it has not seen.

$f(x) = y$

Rote learning

* Training which stores piece of training data & it's classification.

New data classified if it's stored in memory.

Version Space.

Given a set of training examples (positive & negative), the set of hypothesis that correctly map each of the training examples to it's classification is called Version spaces.

Find S algorithm :

Example	Citations	Size	in library	Price	editions	buy
1	some	small	no	affordable	many	no
2	many	big	no	expensive	one	Yes
3	some	big	always	expensive	few	no
4	many	medium	no	expensive	many	Yes
5	many	small	no	affordable	many	Yes

Applying Find S algorithm to find maximmally specific hypothesis

In this data set we have 5 attributes & the target variable is buy.

1. How many concepts are possible for this instance space.

2. How many hypothesis can be expressed by hypothesis language.

3. Apply Find S algorithm on give training set

Ans. 1 $2 \times 3 \times 2 \times 2 \times 3 = 72$

Ans. 2 Here each attribute along with 2 more possibilities are considered ? and NULL.

$4 \times 5 \times 4 \times 4 \times 5 = 1600$

Ans : 3.

step 1 : Initially we initialize hypothesis to more specific value.

$h_0 = \{\phi, \phi, \phi, \phi, \phi\}$

step 2 : $x_1 = \{$some, small, no, affordable, many$\}$ - No

$h_1 = \{\phi, \phi, \phi, \phi, \phi\}$

$x_2 = \{$many, big, no, expensive, one$\}$ - yes

$h_2 = \{$many, big, no, expensive, one$\}$

It is a positive example, here we compare the attribute values in the example & the attribute value in the hypothesis.

If there is a match will keep the same value as it is. & if there is no match will replace the attribute value hypothesis with next most general values.

$x_3 = \{$some, big, always, expensive, few$\}$ - No - ignore.

$h_3 = \{$many, big, no, expensive, one$\}$.

h_3 remains same.

$x_4 = \{$many, medium, no, expensive, many$\}$ - ~~No~~ Yes

$h_4 = \{$many, ~~big~~ ?, no, expensive, ~~one~~ ?$\}$

$x_5 = \{$many, small, no, affordable, many$\}$ - yes.

$h_5 = \{$many, ?, no, ?, ?$\}$.

Decision Tree Induction.

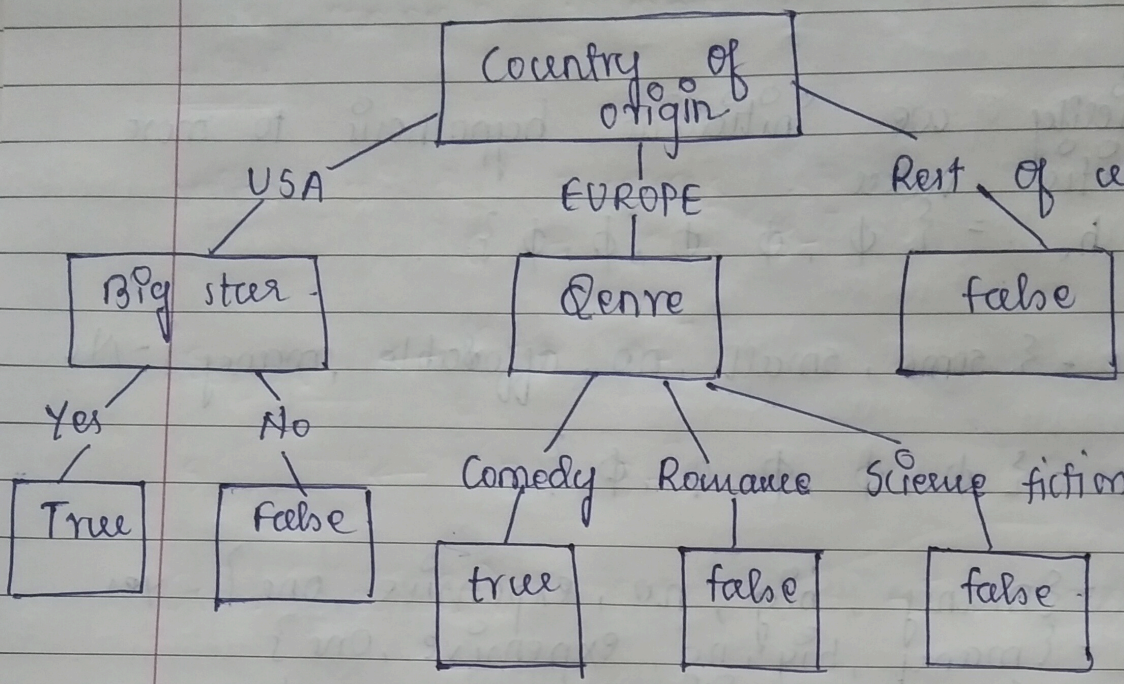


fig: Decision tree determing whether or not film will be by office success.

Entrophy & Information gain.

Day	Outlook	Temp	Humidity	Wind	Play Tennis
D_1	Sunny	Hot	High	weak	No
D_2	Sunny	Hot	High	strong	No
D_3	overcast	Hot	High	weak	Yes
D_4	Rain	mild	Normal	weak	Yes
D_5	Rain	cool	Normal	weak	Yes
D_6	Rain	Cool	Normal	strong	No
D_7	Overcast	cool	High	strong	Yes
D_8	sunny	mild	Normal	weak	No
D_9	sunny	cool	Normal	weak	Yes
D_{10}	Rain	mild	Normal	weak	Yes
D_{11}	Sunny	mild	Normal	strong	Yes
D_{12}	overcast	mild	High	strong	Yes
D_{13}	overcast	Hot	Normal	weak	Yes
D_{14}	Rain	mild	High	strong	No

Entropy : Measures the impurity of collection of examples.

S : collection of training examples.
P_+ : Proportion of +ve example in S
P_- : Proportion of -ve " " S

$$\text{entropy}(S) = -P_+ \log_2 P_+ - P_- \log_2 P_-$$

Characteristics of entropy - Some examples.

$$\text{Entropy}([14+, 0-]) = -\frac{14}{14} \log_2 \left(\frac{14}{14}\right) - 0 \log_2 0 = 0$$

$$\text{Entropy}([9+, 5-]) = -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \left(\frac{5}{14}\right)$$

$$\text{Entropy}(7+, 7-) = \frac{7}{14} \log_2 \left(\frac{7}{14}\right) - \frac{7}{14} \log_2 \left(\frac{7}{14}\right)$$

$$= 1$$

Information Gain.

It measures the expected reduction in entropy caused by partitioning the examples according to an attribute.

Information gain (S, n)

$$\text{Gain}(S, n) = \text{entropy}(S) - \sum_{v \in \text{values}(n)} \frac{|S_v|}{|S|} \text{entropy}(S_v)$$

Example:

To compute information gain of attribute wind.

values (wind) = weak, strong

$S = [9+, 5-]$

$S_{weak} \leftarrow [6+, 2-]$

$S_{strong} \leftarrow (3+, 3-]$

$Gain(S, wind) = Entrophy(S) - \Sigma\, v \in (weak, strong)$

$$Gain(S, wind) = Entrophy(S) - \sum_{v \in (weak, strong)} \frac{|S_v|}{|S|} entrophy(S_v)$$

$$= entrophy(S) - \left(\frac{8}{14}\right) entrophy(S_{weak}) - \left(\frac{6}{14}\right) entrophy(S_{strong})$$

$$= (0.940) - \left(\frac{8}{14}\right)(0.811) - \left(\frac{6}{14}\right)(1.00)$$

$$= 0.048 \longrightarrow wind.$$

Nearest Neighbour algorithm.

Eg: query x = (maths = 6, CS = 8), $k = 3$

	maths	CS	Result
1	4	3	Fail
2	6	7	pass
3	7	8	pass
4	5	5	Fail
5	8	8	pass

To compute we use euclidean distance

$$d = \sqrt{(x_{01} - x_{A1})^2 + (x_{02} - x_{A2})^2}$$

$$= \sqrt{(6-4)^2 + (8-3)^2} = \sqrt{29}$$

$$= 5.38$$

They store the training data & use to determine a classification for each new piece of data as it is encountered.

Learning Neural Network :

Artificial neural network is the network of simple processing nodes modeled on the human brain which achieves it's power through the connectivity between it's neuron each neuron can either fire or not fire by combining millions of neurons together.

Supervise learning networks presented i three classified training data.

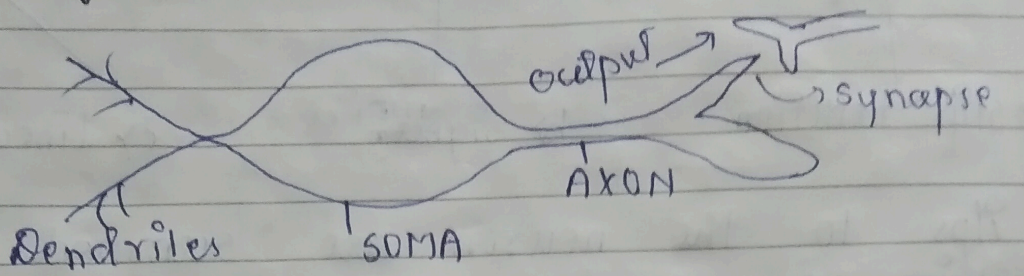
Unsupervise learning methods learn without any human inservention this method is useful in situations where data needs to be classified / clustered into set of classificati-on but classifications are not known in advance

A system that uses reinforcement learning a positive reinforcement given when it perfor-ms correctly & negative reinforcement is given when it performs incorrectly.

Ex: Robotic agent learn by re-inforcement learni-ng about how to pick up an object.

Neurons

i) Biological Neuron



Human brain - neurons
Connected to other neurons - synapse
SOMA

Neuron receives input from other neurons along its dentrits & when this i/p signal exceeds a certain threshold then neuron fires a chemical reaction occurs called as action potensial to the send on to axon.

Artificial Neurons

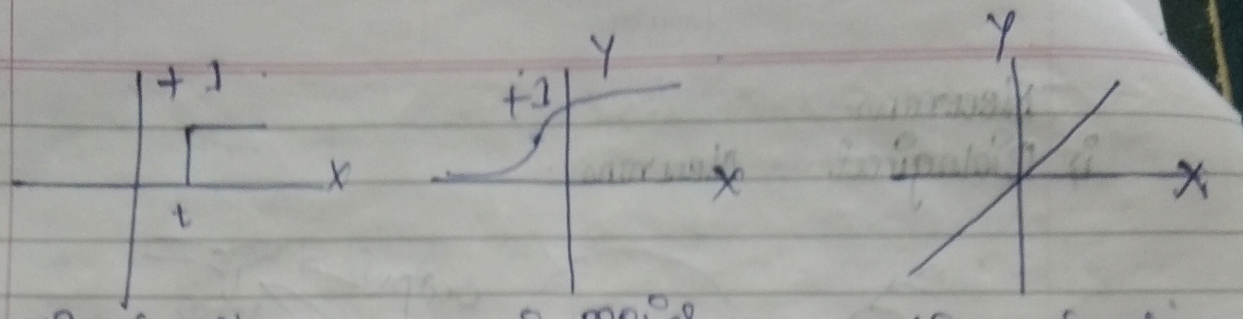
Each node in neural nlw → Receives a number of inputs.

Activation function are applied to the i/p values

Activation level of neuron is o/p value of neuron.

Behaviour of neuron.

$$x = \sum_{i=1} w_i x_i$$



a) Stip function b) Sigmoid c) Linear function.

ilps to the neuron are summed & This summ is compared with the threshold if summ is greater than threshold the neuron fires & has an activation level of +1 otherwise it is inactive & has an activation level of zero.

$$X = \sum_{i=1} w_i x_i$$

Weights

$w_1 = 0.8$

$w_2 = 0.4$

Inputs

$x_1 = 0.7$

$x_2 = 0.9$

Summed weight

$(0.8 \times 0.7) + 0.4 \times 0.9 = 0.92$

Activation level.

$$Y = \begin{cases} +1 & \text{for } x > t \\ 0 & \text{for } x \leq t \end{cases}$$

ROLL NO. DATE

Perceptrons

Perceptron can have any number of input which are arranged into a grid & grid can used to represent an image or field of vision so perceptron can be used to perform some image classification or some recognition tasks

Imp

Perceptrons uses step functions.

+1 if weighted sum of i/p $x >$ threshold t
-1 if $x \leq t$.

$$x = \sum_{i=1}^{n} w_i x_i$$

$$Y = \begin{cases} +1 & \text{for } x > t \\ 0 & \text{for } x \leq t \end{cases}$$

function

$$\text{step}(x) = \begin{cases} +1 & \text{for } x > t \\ 0 & \text{for } x \leq t \end{cases}$$

Activation function for perceptron.

$$Y = \text{step}\left(\sum_{i=0}^{n} w_i x_i\right)$$

Learing process for perceptron.

Random weights are assigned between -0.5 & +0.5

Training data is presented to the perceptron & it's o/p classification is observed If o/p is incorrect the weights are adjusted to try to more closely classify this i/p.

If posatron incorrectly classify a positive piece of training data as negative, then weights need to be modify to increase the o/p for set of i/ps.

$$w_i \leftarrow w_i (\alpha \times x_i \times e)$$

Eg: Logical or function.

Epoch	x_1	x_2	Expected Y	Actual Y	Error	w_1	w_2
1	0	0	0	0	0	-0.2	0.4
1	0	1	1	1	0	-0.2	0.4
1	1	0	1	0	1	0	0.4
1	1	1	1	1	0	0	0.4
2	0	0	0	0	0	0	0.4
2	0	1	1	1	0	0	0.4
2	1	0	1	0	1	0.2	0.4
2	1	1	1	1	0	0.2	0.4

Logical OR function for 2 i/ps $t=0$ & learning rate 0.2.

$\alpha = 0.2$

$w_1 = -0.2 \quad w_2 = 0.4$

$x_1 = 0, \quad x_2 = 0$

∴ expected o/p is $x_1 \vee x_2 = 0$.

we apply formulae for Y

$$Y = \text{step}\left(\sum_{i=0}^{n} w_i x_i\right)$$

$$= \text{step}\ (0 \times -0.2) + (0 \times 0.4)$$

$$= 0$$

o/p Y is expected so weights do not change.

$w_1 = -0.2$ $w_2 = 0.4$

$x_1 = 0$ $x_2 = 1$

∴ expected o/p is $x_1 \vee x_2 = 1$.

we apply formula for Y

$$Y = \text{step}\left(\sum_{i=0}^{n} w_i x_i\right)$$

$$= \text{step}\,(0 \times (-0.2) + (1 \times (0.4))$$

$$= 0.4$$

for $w_1 = -0.2$ $w_2 = 0.4$

$x_1 = 1$ $x_2 = 0$

∴ expected o/p is $x_1 \vee x_2 = 1$.

we apply formula for Y.

$$Y = \text{step}\left(\sum_{i=0}^{n} w_i x_i\right)$$

$$= \text{step}\,(1 \times (-0.2) + (0 \times (0.4)))$$

$$= 0$$

This is incorrect $1 \vee 0 = 1$.

weights are adjusted

so, assign new values to weights.

$$w_1 \leftarrow w_1 + (a \times x_i \times e)$$

$$w_1 = -0.2 + (0.2 \times 1 \times x_1)$$

$$= -0.2 + 0.2$$

$$= 0$$

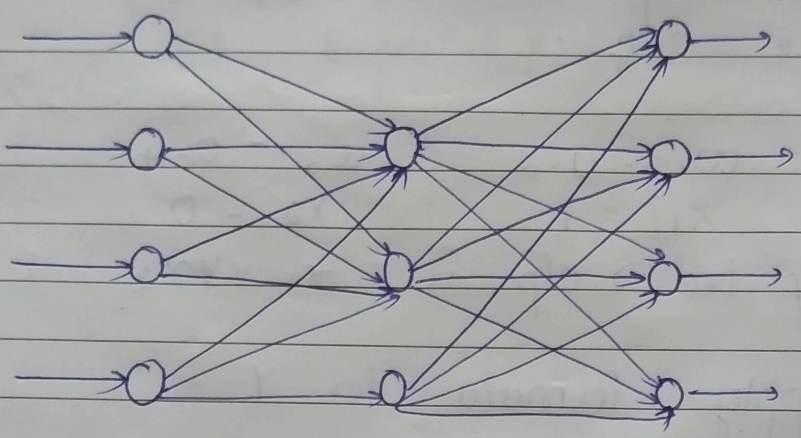
$w_2 = 0.4$

$0.4 + (0.2 \times 0 \times 1)$

for $x_1 = 1$ $x_2 = 1$

$Y = \text{step}(0 \times 1) + (0.4 \times 1)$
$= \text{step}(0) + (0.4)$
$= \text{step}(0.4)$
$= 1$

Multilayer Neural Network.



Three layer field forward neural nlw.

Neural network consist of number of neurons that are connected together. usually arranged in layers.

The above artitectural is neural layer also called as field forward network consist of 3 layers

First layer is Input layer.
In this layer it receives a single i/p signal.,
It passes i/p signals on to nodes in the next layer which is called as middle layer.
The o/p of each node in this layer is passed through each node in final layer which is o/p layer.

ROLL NO. | DATE

If carries final stage of processing & sends o/p sigles

Back propagation.

multilayer n/w = each neuron has weights

when error is made, greater number of weights to be adjusted.

Recurrent Network.

A recurrent network can have connection that go backward from o/p node to i/p node.

Recurrent n/w internal state can alter as set of i/p data presented to it & it can said to have a memory

Ex: Recurrent n/w could be used to Predict the stock market price of particular stock based on all previous values.

Eg: Hopfield Networks

Form of recurrent nlw

Activation function used by Hopfield nlw

$$sign(x) = \begin{cases} +1 & \text{for } x > 0 \\ -1 & \text{for } x < 0 \end{cases}$$

weights of nlw

$$W = \sum_{i=1}^{n} x_i x_i^t - NI$$

Eg: Single layer Hopfield nlw with 5 nodes & 3 training i/ps

$$x_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad x_2 = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \quad x_3 = \begin{bmatrix} 1 \\ -1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

The 3 training states are used to predict the weight matrix will be stable state for the matrix.

We can test this by determining the output vector

$$Y_i = sign(WX_i - \theta).$$

θ is threshold matrix which contains threshold -5 for the each of 5 inputs.

ROLL NO. | DATE

$$W = \sum_{i=1}^{n} X_i^{\circ} X_i^{\circ t} - NI$$

$$= X_1 X_1^t + X_2 X_2^t + X_3 X_3^t - 3I$$

$$= \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

$$+ \begin{bmatrix} 1 \\ -1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & -1 & 1 & 1 & 1 \end{bmatrix} - 3I$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ -1 & 1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} - 3I$$

$$= \begin{bmatrix} 3 & 3 & 3 & 3 & 3 \\ 3 & 3 & 3 & 3 & 3 \\ 3 & 3 & 3 & 3 & 3 \\ 3 & 3 & 3 & 3 & 3 \\ 3 & 3 & 3 & 3 & 3 \end{bmatrix} - \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 0 & 1 & 3 & 3 & 1 \\ 1 & 0 & 1 & 1 & 3 \\ 3 & 1 & 0 & 3 & 1 \\ 3 & 1 & 3 & 0 & 1 \\ 1 & 3 & 1 & 1 & 0 \end{bmatrix}$$

$$Y_1 = \text{sign} \begin{bmatrix} 0 & 1 & 3 & 3 & 1 \\ 1 & 0 & 1 & 1 & 3 \\ 3 & 1 & 0 & 3 & 1 \\ 3 & 1 & 3 & 0 & 1 \\ 1 & 3 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$= \text{sign} \begin{bmatrix} 8 \\ 6 \\ 8 \\ 8 \\ 6 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = X_1$$

$$Y_2 = \text{sign} \begin{bmatrix} 0 & 1 & 3 & 3 & 1 \\ 1 & 0 & 1 & 1 & 3 \\ 3 & 1 & 0 & 3 & 1 \\ 3 & 1 & 3 & 0 & 1 \\ 1 & 3 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$= \text{sign} \begin{bmatrix} 2-8 \\ 4-6 \\ 8-8 \\ 2-8 \\ 4-6 \end{bmatrix}$$

$$\begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} = X_2$$

ROLL NO. DATE

$$Y_4 = \text{sign}\begin{bmatrix} 0 & 1 & 3 & 3 & 1 \\ 1 & 0 & 1 & 1 & 3 \\ 3 & 1 & 0 & 3 & 1 \\ 3 & 1 & 3 & 0 & 1 \\ 1 & 3 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ -1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$= \text{sign}\begin{bmatrix} 2 \\ 2 \\ 2 \\ -4 \\ 2 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

Not in one of the attractors, we need to apply rule again.

$$Y_5 = \text{sign}\begin{bmatrix} 0 & 1 & 3 & 3 & 1 \\ 1 & 0 & 1 & 1 & 3 \\ 3 & 1 & 0 & 3 & 1 \\ 3 & 1 & 3 & 0 & 1 \\ 1 & 3 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$= \text{sign}\begin{bmatrix} 2 \\ 4 \\ 2 \\ 8 \\ 4 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = X_1$$

The Hopfield n/w involves 3 stages.

stage 1: The n/w is trained to learn the attracter state.
This can be thought of as a storage / memorization state.

stage 2: Involves testing in n/w by providing the attracter states as i/p & checking that o/p are identicle.

stage 3: Involves using the n/w which is require to retrieve the data from it's memory.

Unsupervised learning network.

Kohen maps.

1) Self organizing feature map.
2) Trained using unsupervised learning.
3) Used for clustering data.

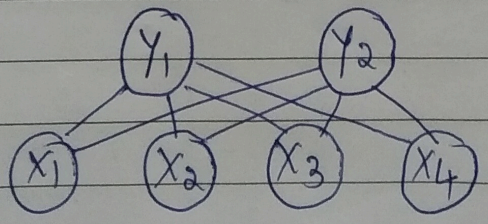
step 1: Initialize weights.

w_{ij}

no. of i/p — no. of cluster.

step 2: Compute the square of eluclidean distance.

$$D(j) = \sum_{i=1}^{n} \sum_{i=1}^{m} (x_i - w_{ij})^2$$



m = 2
j = 1 to 2.

step 3: Find winning index j, D(j) is minimum

step 4: Calculate new weights

$$w_{ij}^{o} = w_{ij}^{o} + \alpha(x_i - w_{ij})$$

step 5: Update learning rate α using the formula

$$\alpha(t+1)$$

step 6: After no. of epochs test for stopping condition of nlw

$w_0 = 1.1234$

$w_0 = 0.1234$

$w_0 = 0.1233$

Eg: x 1) [0 0 1 1]

2) [1 0 0 0]

3) [0 1 1 0]

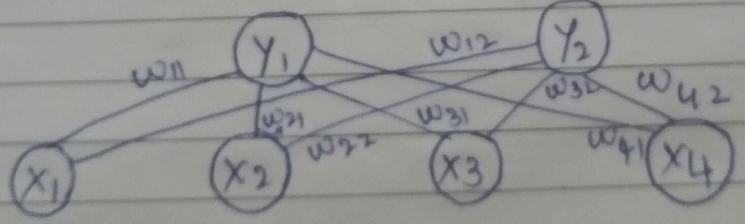
4) [0 0 0 1]

Number of clusters to form 2.

Assume

→ No. of i/p vectors $n = 4$

no. of clusters $m = 2$



Initialize weights randomly bet'n 0 & 1

$$w.j = \begin{bmatrix} 0.2 & 0.9 \\ 0.4 & 0.7 \\ 0.6 & 0.5 \\ 0.8 & 0.3 \end{bmatrix}$$

weights going for cluster 1: w_{i1} w_{i2} → cluster 2

First: i/p vector $x = [0 \quad 0 \quad 1 \quad 1]$

Calculate euclidean distance

$$D(j) = \sum_{i=1}^{n} \sum_{j=1}^{m} (x_i - w_{ij})^2$$

$$D(1) = (x_1 - w_{11})^2 + (x_2 - w_{21})^2 + (x_3 - w_{31})^2 + (x_4 - w_{41})^2$$

$$= (0 - 0.2)^2 + (0 - 0.4)^2 + (1 - 0.6)^2 + (1 - 0.8)^2$$

$$= 0.04 + 0.16 + 0.16 + 0.04$$

$$= 0.4$$

$$D(2) = (x_1 - w_{12})^2 + (x_2 - w_{22})^2 + (x_3 - w_{32})^2 + (x_4 - w_{42})^2$$

$$= (0 - 0.9)^2 + (0 - 0.7)^2 + (1 - 0.5)^2 + (1 - 0.3)^2$$

$$= 2.04$$

$D_1 < D_2$

ROLL NO. | DATE

$$w_{ij}^{00} = w_{ij}^{00} + \alpha (x_i^0 - w_{ij}^{00}).$$

$$w_{11} = w_{11} + 0.5 (0 - 0.2) = 0.1$$

$$w_{21} = w_{21} + \alpha (x_2 - w_{21})$$

$$= 0.2.$$

$$w_{31} = 0.8$$

$$w_{41} = 0.9.$$

$$w_{ij}^{00} = \begin{bmatrix} 0.1 & 0.9 \\ 0.2 & 0.7 \\ 0.8 & 0.5 \\ 0.9 & 0.3 \end{bmatrix}$$

Second i/p vector