Name: Srushti Pravin Karadi
PRN: 22610005

M T W T F S S
Page No.:
Date:
YOUVA

ANN
Assignment 2

Q1) Write the algorithm steps for perceptron learning rule.

→ 1) Initialize weights (w) and bias 'b' to small random values or zero, set the learning rate η.

2) Repeat with convergence:
For each training sample: (x, d)
- compute net 'input' ⇒ $wx+b$
- Apply activation function: $y=1$ if net ≥ 0 else $y=-1$
- compute error: $e = d - y$, $(di - oi)$
- update weight and bias:
- $w = w + \eta e x$, $b = b + \eta e$

3) Stop when all sample are correctly classified or max iteration reached.

4) Use trained perceptron for classification with updated weight vector.

Q2) Derive the weight update equation for delta learning rule.

→ Step ①:
Least squared error between oi and di
Calculating the gradient vector w.r.t. wi of squared error.

$$E \triangleq \frac{1}{2}(di - oi)^2$$

d_i = desired output
o_i = actual output

Step ②:
compute the gradient to minimize error.

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i}\left(\frac{1}{2}(d_i - o_i)^2\right)$$

$$\nabla E = \frac{\partial}{\partial w_i}\left[\frac{1}{2}(d_i - f(w_i^t x))^2\right]$$

using the chain rule for differentiation,

$$\nabla E = \frac{\partial E}{\partial f(w_i^t x)} \cdot \frac{\partial f(w_i^t x)}{\partial (w_i^t x)} \cdot \frac{\partial (w_i^t x)}{\partial w_i}$$

1) calculating:

$$\frac{\partial E}{\partial f(w_i^t x)} = \frac{\partial}{\partial f(w_i^t x)} \frac{(d_i - f(w_i^t x))^2}{2}$$

$$= \frac{1}{2} \times 2 (d_i - f(w_i^t x))(0-1)$$

$$= -[d_i - f(w_i^t x)]$$

2) $$\frac{\partial f(w_i^t x)}{\partial (w_i^t x)} = f'(w_i^t x)$$

3) $$\frac{\partial (w_i^t x)}{\partial w_i} = x$$ (its transpose and not power)

put values,

$$\nabla E = -[d_i - f(w_i^t x)] f'(w_i^t x)\, x \longrightarrow \text{Error gradient } \nabla E$$

Step ③:
since the minimization of the error requires the weight changes to be in the negative gradient direction, we take;

$\Delta w_i = \eta \Delta E$

$\Delta w = \eta \times$ learning signal $\times x$ ← deduced from observation

$\Delta w = -\eta \Delta E$

Substitute

$\therefore \Delta w = -\eta [-(d_i - o_i) f'(w_i^t x) x)]$

$\therefore \Delta w = \eta (d_i - o_i) f'(w_i^t x) x$ ← Hence proved which is change in weight.

new weight w

$w_{new} = w_{old} - \eta \nabla E$

$\therefore w_{new} = w_{old} + \Delta w$

Q3) Explain the Hebbian learning rule and widrow Hoff learning rule.

→ Hebbian learning rule:

- proposed by Donald Hebbian 1949.
- inspired by neurobiological principles 'cells that fire together, wire together'
- Unsupervised learning rule where it does not require labelled data.

$$\boxed{\Delta w = c \cdot f(w^t x) x}$$

↑ Learning signal.

- The rule states that if the cross product of o/p and i/p or correlation term of x_i is positive this results in an increase of weight w_{ij}, otherwise the weight decreases. It can be seen that the o/p is strengthened in term for each input.

presented. Hence, frequent input patterns will have most influence at neurons weight vector.

Widrow-Hoff learning Rule
(least mean squares) LMS learning rule (1962).

- Applied for supervised learning of neural N/W.
- Independent of Activation function as it minimizes the squared error between the desired o/p (di) and neuron activation value i.e. $net_i = w_i^t x$.
- Learning signal for this value rule is given as,

$$r \triangleq d_i - w_i^t x$$

The weight vector increment under this rule is,

$$\Delta w_i = c \cdot (d_i - w_i^t x) x$$

for signal weight adjustment

$$\Delta w_{ij} = c(d_i - w_i^t x)\, x_j, \quad j = 1, 2, \ldots n$$

- can be considered special case of delta learning rule.

Q4) Compare all learning rules w.r.t. weight adjustment, initial weight and types of learning.

Learning rule	Weight Adjustment	Initial Weights	Type of Learning	Error Signal	Activation function
Perceptron	$\Delta w_j = c(d_i - o_i)x$	small random values or zeros	Super-vised	$d_i - o_i$ (binary error)	step function
Hebbian	$\Delta w_j = c x_i o_j$	small random values or zeros	Un-super-vised	None (activation based)	None (linear)
Widraw Hoff	$\Delta w = c(d_i - w^T x)x$	-11-	Super-vised	$d - w^T x$ (linear error)	Linear.
Correlation	$\Delta w = cdx$	-11-	Super-vised	d (target o/p)	Linear
Winner Take all	$\Delta w_{ij} = c x_i o_j$ (only for winning neuron)	-11-	Unsuper-vised	Non (competitive learning)	competitive (max function)
Delta	$\Delta w = c(d-o) f'(w^T x) x$	-11-	Super-vised	$d - o$	Non Linear (sigmoid tanh)

Q5) Train a NN using Perceptron learning rule.

$x_1 = [-1 \quad 3 \quad -2 \quad 1] \quad d_1 = 1$

$x_2 = [-1 \quad 2 \quad -1 \quad 0] \quad d_2 = -1$

$x_3 = [-1 \quad -3 \quad 0 \quad -2] \quad d_3 = 1$

$w = [0.2 \quad 0.4 \quad 1 \quad 0] \quad c = 1$

22610005

→ ① consider x_1 with $d_1 = 1$

$net_1 = w^T x_1$

$= \begin{bmatrix} 0.2 & 0.4 & 1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 3 \\ -2 \\ 1 \end{bmatrix} = -0.2 + 1.2 - 2 + 0$

$= -1$

$f(net_1) = sign(net_1) = sign(-1) = -1$

$\therefore d_1 \neq o_1$

Hence, updating the weight vector,

$w_1 = w + c(d_i - o_i) x_1$

$= \begin{bmatrix} 0.2 \\ 0.4 \\ 1 \\ 0 \end{bmatrix} + 1 (1 - (-1)) \begin{bmatrix} -1 \\ 3 \\ -2 \\ 1 \end{bmatrix}$

$w_1 = \begin{bmatrix} -1.8 \\ 6.4 \\ -3 \\ 2 \end{bmatrix}$

② $net_2 = \begin{bmatrix} -1.8 & 6.4 & -3 & 2 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \\ -1 \\ 0 \end{bmatrix} = 1.8 + 2.8 + 3$

$= 17.6$

$f(net_1) = 1 \quad d_2 = -1$

$\therefore w_2 = \begin{bmatrix} -1.8 \\ 6.4 \\ -3 \\ 2 \end{bmatrix} + 1 (-1 - 1) \begin{bmatrix} -1 \\ 2 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1.8 + 2 \\ 6.4 - 4 \\ -3 + 2 \\ 2 - 0 \end{bmatrix} = \begin{bmatrix} 0.2 \\ 2.4 \\ -1 \\ 2 \end{bmatrix}$

$w_2 = \begin{bmatrix} 0.2 \\ 2.4 \\ -1 \\ 2 \end{bmatrix}$

22610005

③ $net_3 = \begin{bmatrix} 0.2 & 2.4 & -1 & 2 \end{bmatrix} \begin{bmatrix} -1 \\ -3 \\ 0 \\ -2 \end{bmatrix} = -0.2 - 7.2 - 4$

$f(net_3) = -1$

$d_3 = +1$

$\therefore w_3 = \begin{bmatrix} 0.2 \\ 2.4 \\ -1 \\ 2 \end{bmatrix} + 1(1+1) \begin{bmatrix} -1 \\ -3 \\ 0 \\ -2 \end{bmatrix} = \begin{bmatrix} 0.2-2 \\ 2.4-6 \\ -1 \\ 2-4 \end{bmatrix} = \begin{bmatrix} -1.8 \\ -3.6 \\ -1 \\ -2 \end{bmatrix}$

Hence found the updated weight vectors.

Q.6) Implement the perceptron rule training of N/w from figure using $f(net) = sgn(net)$.

$c = 1$

$w_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \left(x_1 = \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix}, d_1 = -1 \right) \left(x_2 = \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}, d_2 = 1 \right)$

Repeat the training sequence $(x_1\ d_1)$, $(x_2\ d_2)$ until two correct responses in a row are achieved.

→ ① $net_1 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix} = 1$

$f(net_1) = 1, \quad d_1 = -1$

$\therefore w_2 = w_1 + c(d_i - o_i)x_i$

$= \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + 1(-1-1) \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -4 \\ -1 \\ 2 \end{bmatrix}$

② $net_2 = w_2^T x_2$

$= \begin{bmatrix} -4 & -1 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix} = 0 + 1 - 2 = -1$

$f(net_2) = -1 \quad d_2 = 1$

$\therefore \quad w_3 = w_2 + c(d_2 - o_2)\, x_2$

$= \begin{bmatrix} -4 \\ -1 \\ 2 \end{bmatrix} + 1\,(1-(-1)) \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix} = \begin{bmatrix} -4 \\ -3 \\ 0 \end{bmatrix}$

Q8) Train a NN using widrow Hoff rule (LMS)

$x_1 = [-1 \;\; 0 \;\; 2] \quad d_1 = 1$

$x_2 = [-1 \;\; -2 \;\; 1] \quad d_2 = -1$

$c = 0.25, \quad w_1 = [1 \;\; 0 \;\; 1]$

→ ① $r = d_i - o_i$

$\therefore \quad \Delta w = c(d_i - o_i)\, x$ ... independent of activation function

$w_2 = w_1 + c(d_1 - o_1)\, x_1$

$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + 0.25\,(1 - [1 \;\; 0 \;\; 1] \begin{bmatrix} -1 \\ 0 \\ 2 \end{bmatrix}) \begin{bmatrix} -1 \\ 0 \\ 2 \end{bmatrix}$

$\therefore \quad w_2 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$

② $w_3 = w_2 + c(d_2 - o_2)\, x_2$

$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + 0.25\,(-1 - [1 \;\; 0 \;\; 1] \begin{bmatrix} -1 \\ -2 \\ 1 \end{bmatrix}) \begin{bmatrix} -1 \\ -2 \\ 1 \end{bmatrix}$

$= -11- + 0.25\,(-1 - (-1+1)) \begin{bmatrix} -1 \\ -2 \\ 1 \end{bmatrix}$

$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0.25 \\ 0.50 \\ -0.25 \end{bmatrix}$

$\therefore \quad w_3 = \begin{bmatrix} 1.25 \\ 0.50 \\ 0.75 \end{bmatrix}$

Q9) Train same N/w using correlation learning rule.

→ For correlation learning rule,

$\Delta W = cdx$

$w_1 = [1 \quad 0 \quad 1]$

$w_2 = w_1 + \Delta W$

$$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + 0.25(1) \begin{bmatrix} -1 \\ 0 \\ 2 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -0.25 \\ 0 \\ 0.50 \end{bmatrix}$$

$$w_2 = \begin{bmatrix} 0.75 \\ 0 \\ 1.5 \end{bmatrix}$$

$w_3 = w_2 + \Delta W$

$$= \begin{bmatrix} 0.75 \\ 0 \\ 1.5 \end{bmatrix} + 0.25(-1) \begin{bmatrix} -1 \\ -2 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.75 \\ 0 \\ 1.5 \end{bmatrix} + \begin{bmatrix} 0.25 \\ 0.50 \\ -0.25 \end{bmatrix}$$

$$w_3 = \begin{bmatrix} 1 \\ 0.5 \\ 1.25 \end{bmatrix}$$

Q10) Train a neural n/w using delta learning rule.

$x_1 = [1 \quad -2 \quad 0 \quad -1]$ $\quad d_1 = 1$

$x_2 = [0 \quad 1.5 \quad -0.5 \quad -1]$ $\quad d_2 = -1$

$x_3 = [-1 \quad 1 \quad 0.5 \quad -1]$ $\quad d_3 = 1$

$w_1 = [1 \quad -1 \quad 0 \quad 0.5]$

$c = 1, \quad \lambda = 1$

2261005

Page No.: Date: YOUVA

→ ① $net_1 = w^T x_1 = [1 \quad -1 \quad 0 \quad 0.5] \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} = -1.5$

$$f(net_1) = \frac{2}{1+e^{-net_1}} - 1 = \frac{2}{1+e^{1.5}} - 1 = -0.6351$$

$$f'(net_1) = \frac{1}{2}(net_1)^2 = \frac{1}{2}(-0.6351)^2 = 0.2017$$

$$w_2 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + 1\,(1-(-0.6351))\,0.2017 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + 0.3298 \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0.3298 \\ -1.6596 \\ 0 \\ 0.1702 \end{bmatrix}$$

② $net_2 = w^T x_2 = [1.3298 \quad -1.6596 \quad 0 \quad 0.1702] \begin{bmatrix} 0 \\ +1.5 \\ -0.5 \\ -1 \end{bmatrix}$

$$= -1$$

$$f(net_2) = \frac{2}{1+e^{-net_2}} - 1 = \frac{2}{1+e^{1}} - 1 = -0.4621$$

$$f'(net_2) = \frac{1}{2}(1)^2 = \frac{1}{2} = 0.5$$

$$w_3 = \begin{bmatrix} 1.3298 \\ -1.6596 \\ 0 \\ 0.1702 \end{bmatrix} + 1\,(-1-(-0.4621))\,(0.5) \begin{bmatrix} 0 \\ 1.5 \\ 0.5 \\ -1 \end{bmatrix}$$

2261D005

M T W T F S S
Page No.:
Date:
YOUVA

$\therefore$ $w_3 = \begin{bmatrix} 1.3298 \\ -2.0631 \\ -6.1345 \\ 6.4392 \end{bmatrix}$

③ $net_3 = w_3^T x_3 = \begin{bmatrix} -1.3298 & -2.0631 & -6.1345 & 0.4392 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$

$= -3.8994$

$f(net_3) = \frac{2}{1+e^{-net_3}} - 1 = -0.9603$

$f'(net_3) = \frac{1}{2}(1 - (0.9603))^2 = 1.9214$

$w_4 = \begin{bmatrix} 1.3298 \\ -2.0631 \\ -0.1345 \\ 6.4392 \end{bmatrix} + 1(1+0.9603)1.9214 \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$

$w_4 = \begin{bmatrix} -2.4367 \\ 1.7034 \\ 1.7488 \\ -3.3278 \end{bmatrix}$