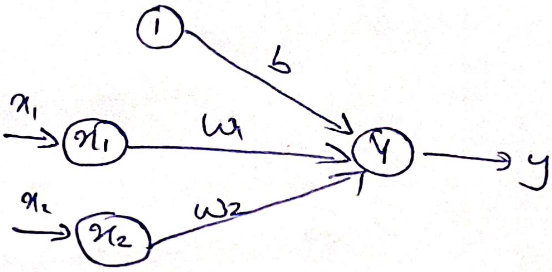
Soft Computing

- Implement OR function using the perceptron training algorithm with binary input and bipolar target



Ans Truth Table:

x_1	x_2	t
1	1	1
1	-1	1
-1	1	1
-1	-1	-1

Initial weight $w_1 = w_2 = 0$ $b = 0$ $\alpha = 1$

$$y = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

EPOCH-1

where; $y_{in} = w_1 x_1 + w_2 x_2 + b$

Pattern 1: $x_1 = 1$ $x_2 = 1$ $t = 1$

$Y_{in} = (\overset{x_1}{1} \times \overset{w_1}{0}) + (\overset{x_2}{1} \times \overset{w_2}{0}) + 0 = 0$ $\quad Y_{in} = 0 \Rightarrow Y = 0$

$t = 1$ $Y = 0$: error: update required ($t \neq Y$)

$w_1(new) = w_1(old) + \alpha t x_1$

$w_1(new) = 0 + (1 \times 1 \times 1) = 1$

$w_2(new) = 0 + (1 \times 1 \times 1) = 1$

$b(new) = 0 + 1 \times 1 = 1$ $\quad b(new) = b(old) + \alpha t$

Pattern 2 ($x_1 = 1$ $x_2 = -1$ $t = 1$)

$Y_{in} = (1 \times 1) + (-1 \times 1) + 1 = 1$

$Y_{in} > 0 \Rightarrow Y = 1$

$t = 1$, $Y = 1$ $t = Y$ (no update) $\quad w_1 = w_2 = b = 1$

Pattern 3: $x_1 = -1$, $x_2 = 1$, $t = 1$

$Y = (-1 \times 1) + (1 \times 1) + 1 = 1$

$Y = 1$, $t = 1$, $Y = t$ (no update)

Pattern 4: ($x_1 = -1$, $x_2 = -1$, $t = -1$)

$Y_{in} = (-1 \times 1) + (-1 \times 1) + 1 = -1$

$Y_{in} < 0 \Rightarrow Y = -1$

$t = -1$, $Y = -1$ $t = Y$ so no update

EPOCH · 2

All output are correct, So no wt change. Training stop

final weight : $w_1 = 1$, $w_2 = 1$, $b = 1$

2. Explain the Training algorithm used for a perceptron network with single output classes

The Perceptron algorithm proposed by frank Rosenblatt is a supervised learning Algo used to train single layered perceptron

Steps in Training:

step0: Initialization:

Initialize all wt to small random value or zero

Initialize bias b and also choose learning rate α $(0 < \alpha \le 1)$

step1: Perform step 2-6 until final stopping condition is false

Step2: Perform step 3-5 for each training pair indicated by s:t

Step3: The i/p layer contain input unit is applied with identity activation f^n $x_i = s_i$

Step4: Calculate the o/p of network

$$Y_{in} = b + \sum_{i=1}^{n} x_i w_i \qquad Y = f(Y_{in}) = \begin{cases} 1 & \text{if } Y_{in} \ge \theta \\ 0 & \text{if } -\theta \le Y_{in} \le \theta \\ -1 & \text{if } Y_{in} < -\theta \end{cases}$$

step5: weight and bias adjustment

If $Y \ne t$ $w_{(new)} = w_{(old)} + \alpha t x_i$, $b_{(new)} = b_{(old)} + \alpha t$

else; $w_{new} = w_{old}$, $b_{(new)} = b_{(old)}$

Step6: Train the network until there is not wt change, else start from 2.