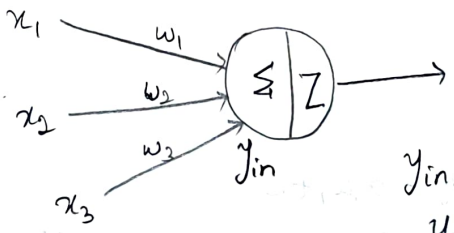
Assignment - 1 :

Mc Culloch - Pits (MCP) Neuron Model :

1) Give the Architecture of the MCP neural network model.

The MCP Neural Network Model refers to the Mc Culloch - pitts Neural Network, which is one of the earlist models of artificial neurons. The neurons are connected by directed weighted paths. McCulloch - pitt neuron allows binary activation, it either fires with an activation 1 or does not fire with an activation of 0. If $w > 0$, then the connected path is said to be excitatory else it is known as Inhibitory.

Excitatory connections have positive weights and inhibitory connections have negative weights. Each neuron has a fixed threshold for firing. If the net input to the neuron is greater than the threshold, it fires.



$$y_{in} = x_1w_1 + x_2w_2 + \dots + x_nw_n$$

$$y_{in} = \sum_{i=1}^{n} x_iw_i$$

Activation function $= f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq \theta \\ 0 & \text{if } y_{in} < \theta \end{cases}$ θ = threshold

(i) input layer :- A set of Binary inputs $x_1, x_2, x_3 \dots x_n$ Each input represents a feature of variable and takes a binary value 0 & 1.

(ii) Weights :- Each input x is associated with a weight which represents the importance of the input. The weight sum of inputs is calculated as $y_{in} = \sum_{i=1}^{n} x_iw_i$

(iii) Summarization function :- Computes the linear combination of weighted sum of inputs.

(iv) Threshold Function :- A threshold θ is applied to the weighted sum y_{in}. If $y_{in} \geq \theta$ the neuron fires (output = 1). If $y_{in} < T$, the neuron doesnot fire (Output = 0).

$$y_{out} = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq \theta \\ 0 & \text{if } y_{in} < \theta \end{cases}$$

(v) Output :- A single Binary output y_{out}, which represents the activation state of the neuron.

2) Generate the output of logic NOT, NAND, and NOR function using MCP neuron.

NOT function

Input	output	$y_{in} = x_1 w_1 + b$
0	1	0
1	0	1

$w_1 = 1 \quad b = 0$

$x_1 = 0 \quad y_{in_1} = 1 \times 0 + 0 = 0$

$x_2 = 1 \quad y_{in_2} = 1 \times 1 + 0 = 1$

Activation function $= y_{out} = f(y_{in}) = \begin{cases} 0 & \text{if } y_{in} \geq 1 \\ 1 & \text{if } y_{in} < 1 \end{cases}$

threshold $= \theta = 1$

$y_{out1} = f(y_{in_1}) = f(0) = 1$

$y_{out2} = f(y_{in_2}) = f(1) = 0$

Nand Function

x_1	x_2	Nand (x_1, x_2)	$y_{in} = x_1 + x_2$	y_{out}
0	0	1	0	1
0	1	1	1	1
1	0	1	1	1
1	1	0	2	0

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

$w_1 = w_2 = 1$, $b = 0$

threshold $= \theta = 1$

Activation function $= \begin{cases} 1 & y_{in} < 2 \\ 0 & y_{in} \geq 2 \end{cases}$

$x_1 = 0 \quad x_2 = 0 \Rightarrow y_{in} = (1 \times 0) + (0 \times 1) = 0$

$x_1 = 0 \quad x_2 = 1 \Rightarrow y_{in} = (0 \times 1) + (1 \times 1) = 1$

$x_1 = 1 \quad x_2 = 0 \Rightarrow y_{in} = (1 \times 1) + (0 \times 1) = 1$

$x_1 = 1 \quad x_2 = 1 \Rightarrow y_{in} = (1 \times 1) + (1 \times 1) = 2$

NOR Function

x_1	x_2	$y_{in} = x_1 w_1 + x_2 w_2 + b$	NOR (x_1, x_2)
0	0	0	1
0	1	1	0
1	0	1	0
1	1	2	0

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

$w_1 = 1 \quad w_2 = 1 \quad b = 0 \quad \theta = 1$

$x_1 = 0 \quad x_2 = 0 \Rightarrow y_{in} = (0 \times 0) + (1 \times 0) = 0$

$x_1 = 0 \quad x_2 = 1 \Rightarrow y_{in} = (1 \times 0) + (1 \times 1) = 1$

$x_1 = 1 \quad x_2 = 0 \Rightarrow y_{in} = (1 \times 1) + (0 \times 1) = 1$

$x_1 = 1 \quad x_2 = 1 \Rightarrow y_{in} = (1 \times 1) + (1 \times 1) = 2$

Activation function

$y_{out} = f(y_{in})$

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

3) Write the Limitations of MCP neuron Model

* Binary Output :- MCP neuron produces only binary output (0 or 1), which is overly simplistic. It cannot handle continuous or multi-valued inputs and outputs, which are essential for real-world problems.

* Linear Speable Problems only :- MCP model can only solve problems that are linearly seperable (AND, OR).
If fails to solve problems like XOR, which require non-linear separability.

* No Learning Mechanism :- The weights and thresholds in the MCP model are manually assigned to and do not changes
It lacks the ability to learn from data through process, which is critical feature of modern neural networks.

* No support for Non-primary logic :- The model cannot represent or process fuzzy logic & probabilistic outputs, which as important for handling uncertainity and ambiguity.

* Static Architecture :- The MCP model has fixed structure with no provision for adding hidden layers or neurons.
Modern neural networks leverage multi-layer architecture to capture complex patterns, which the MCP model cannot achieve.

* No activation function flexibility :- The MCP neuron uses a simple step function for activation, limiting its ability to model complex relationships between inputs and outputs. It cannot use modern activation functions like sigmoid, ReLu & softmax, which enable richer functionality.

* Limited practical applicability :- Due to its simplicity, the MCP model cannot address modern machine learning tasks like image recognition, natural language processing & speech synthesis

* Lack of feedback and Recurrent Connections :-
The MCP model is a feed foward network, which means information flows in one direction only.
It cannot model temporal & sequential data, which requires recurrent & feedback connections.

ASSIGNMENT - 2

PERCEPTRON MODELS

Develop a perception for the following logic functions with bipolar inputs and targets (intial weights are 0, learning rate = 1, and threshold = 1)

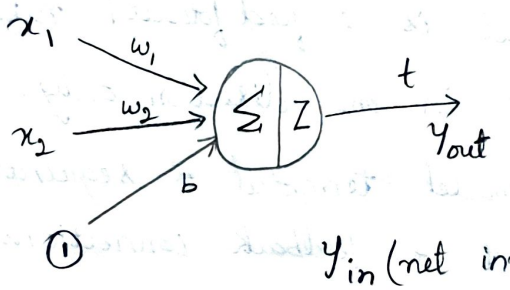
(i) OR

(ii) NAND

(iii) NOR

(iv) NOT

(i) OR



y_{in} (net input) $= x_1 w_1 + x_2 w_2 + b$

Given,

$w_1 = w_2 = b = 0$, $\alpha = 1$, $\theta = 1$

x_1	x_2	$t = x_1$ OR x_2	$y_{out} = ?$
1	1	1	?
1	-1	1	
-1	1	1	
-1	-1	-1	

Activation function $y_{out} = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 1 \\ -1 & \text{if } y_{in} < 1 \end{cases}$

① For first inputs -

$x_1 = 1$ $x_2 = 1$ $t = 1$

$w_1 = 0$ $w_2 = 0$ $b = 0$

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

$$y_{in} = 1 \times 0 + 1 \times 0 + 0$$

$$y_{in} = 0$$

$$y_{out} = f(y_{in})$$

$$y_{out} = f(0) = -1$$

$$t = 1 \neq y_{out} = -1$$

→ update the weights

$$w_1(n) = w_1(0) + \alpha t x_1$$

$$w_1(n) = 10 + 1 \times 1 \times 1$$

$$w_1(n) = 1$$

$$w_2(n) = w_2(0) + \alpha t x_2$$

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

$$w_2(n) = 1$$

$$b(n) = b(0) + \alpha t$$

$$= 0 + 1 \times 1$$

$$b(n) = 1$$

→ $x_1 = 1 \quad x_2 = 1 \quad t = 1$

$w_1 = 1 \quad w_2 = 1 \quad b = 1$

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

$$= 1 \times 1 + 1 \times 1 + 1 \times 1$$

$$y_{in} = 3$$

$$y_{out} = f(y_{in}) = f(3)$$

$$y_{out} = 1$$

$$(target = 1) == (y_{out} = 1)$$

② For 2nd set of inputs

$x_1 = 1 \quad x_2 = -1 \quad t = 1$

$w_1 = 1 \quad w_2 = 1 \quad b = 1$

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

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

$y_{in} = 1$

$y_{out} = f(y_{in}) = f(1)$

$y_{out} = 1$

$(target = 1) == (y_{out} = 1)$

③ For 3rd set of inputs

$x_1 = -1 \quad x_2 = 1 \quad t = 1$

$w_1 = 1 \quad w_2 = 1 \quad b = 1$

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

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

$y_{in} = 1$

$y_{out} = f(y_{in}) = f(1)$

$y_{out} = 1$

$(target == 1) == (y_{out} = 1)$

④ For 4th set of inputs

$x_1 = -1 \quad x_2 = -1 \quad t = -1$

$w_1 = 1 \quad w_2 = 1 \quad b = 1$

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

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

$y_{in} = -1$

$y_{out} = f(y_{in}) = f(-1) = -1$

(target = -1) == (Yout = -1)

For the linear class - unseen class

$$w_1x_1 + w_2x_2 + b = 0$$

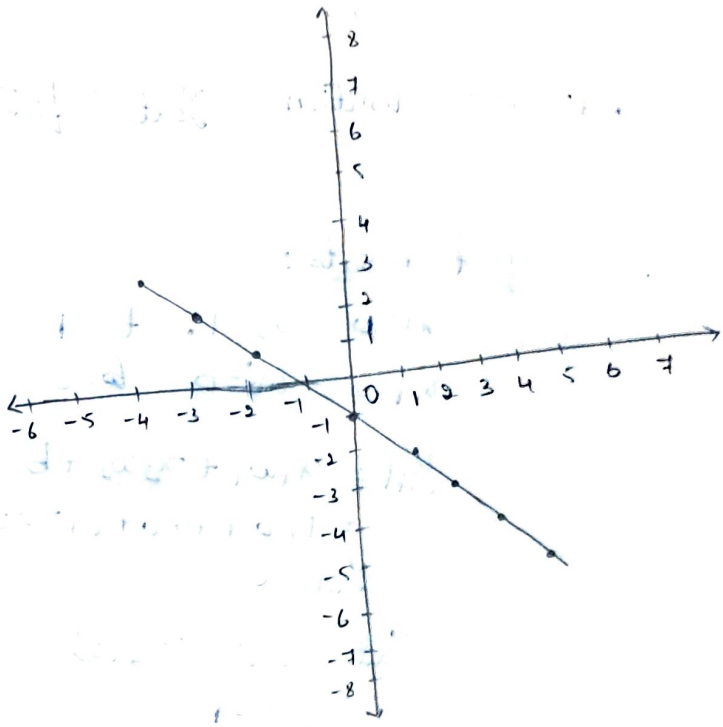
$$w_1 = 1 \quad w_2 = 1 \quad b = 1$$

$$x_1 \times 1 + x_2 \times 1 + 1 = 0$$

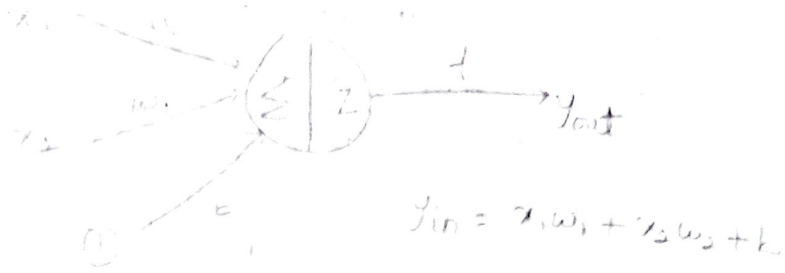
$$x_1 = -x_2 - 1$$

x_1	x_2	$t = x_1$ OR x_2	Y_{out}
1	1	1	1
1	-1	1	1
-1	1	1	1
-1	-1	-1	-1

x_1	$x_2 = -x_1 - 1$
-4	$-(-4)-1 = +3$
-3	2
-2	1
-1	0
0	-1
1	-2
2	-3
3	-4
4	-5



(10) NAND



Given $w_1 = w_2 = b = 0$, $\alpha = 1$ $\theta = 0$

x_1	x_2	$t = x_1$ NAND x_2	$y_{out} = ?$
1	1	-1	
1	-1	1	
-1	1	1	?
-1	-1	1	

Activation function $y_{out} = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$

① For first inputs:

$x_1 = 1$ $x_2 = 1$ $t = -1$

$w_1 = 0$ $w_2 = 0$ $b = 0$

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

$= 1 \times 0 + 1 \times 0 + (-1) \times 0$

$y_{in} = 0$

$y_{out} = f(y_{in}) = f(0)$

$y_{out} = -1$

$(target = -1) == (y_{out} == -1)$

② For second inputs:

$x_1 = 1$ $x_2 = -1$ $t = 1$

$w_1 = 0$ $w_2 = 0$ $b = 0$

① For first inputs

$x_1 = 1$ $x_2 = 1$ $t = -1$

$w_1 = 0$ $w_2 = 0$ $b = 0$

$y_{in} = 0$

$y_{out} = f(y_{in}) = f(0)$

$y_{out} = 1$

$t = -1 \neq y_{out} = 1$

$w_1(n) = w_1(0) + \alpha t x_1$

$w_1(n) = 0 + 1 \times -1 \times 1$

$w_1(n) = -1$

$w_2(n) = w_2(0) + \alpha t x_2$

$w_2(n) = 0 + 1 \times -1 \times 1$

$w_2(n) = -1$

$b(n) = b(0) + \alpha t$

$b(n) = 0 + 1 \times -1$

$b(n) = -1$

$y_{in} = x_1w_1 + x_2w_2 + b$

$= 1\times0 + (-1)\times0 + 0$

$y_{in} = 0$

$y_{out} = f(y_{in}) = f(0)$

$y_{out} = -1$

$target = 1 \neq y_{out} = -1$

→ update the weigths

$w_1(n) = w_1(0) + \alpha t x_1$

$w_1(n) = 0 + 1\times1\times1$

$w_1(n) = 1$

$w_2(n) = w_2(0) + \alpha t x_2$

$w_2(n) = 0 + 1\times1\times-1$

$w_2(n) = -1$

$b_{(n)} = b_{(0)} + \alpha t$

$= 0 + 1\times1$

$b_{(n)} = 1$

→ $x_1 = 1 \quad x_2 = -1 \quad t = 1$

$w_1 = 1 \quad w_2 = -1 \quad b = 1$

$y_{in} = x_1w_1 + x_2w_2 + b$

$= (1\times1) + (-1\times-1) + 1$

$y_{in} = 3$

$y_{out} = f(y_{in}) = f(3)$

$y_{out} = 1$

$(target = 1) == (y_{out} = 1)$

③ For third inputs:

$x_1 = -1 \quad x_2 = 1 \quad t = 1$

$w_1 = 1 \quad w_2 = -1 \quad b = 1$

$y_{in} = x_1w_1 + x_2w_2 + b$

$y_{in} = -1$

$y_{out} = f(y_{in}) = -1$

$y_{in} = x_1w_1 + x_2w_2 + b$

$y_{in} = -1\times1 + (1\times-1) + (-1)$

$y_{in} = -3$

$(target = -1) == (y_{out} = -1)$

② for second inputs

$x_1 = 1 \quad x_2 = -1 \quad t = 1$

$w_1 = -1 \quad w_2 = -1 \quad b = -1$

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

$y_{in} = -1$

$y_{out} = -1$

$(target = 1) \neq (y_{out} = -1)$

$w_1(n) = -1 + (1\times1\times1)$

$w_1(n) = 0$

$w_2(n) = -1 + (-1\times1\times1)$

$w_2(n) = -2$

$b_{(n)} = -1 + (1\times1)$

$b_{(n)} = 0$

$y_{in} = 0 + (-2\times-1) + 0$

$y_{in} = 2, \quad y_{out} = 1$

$(target = 1) = (y_{out} = 1)$

③ for third inputs

$x_1 = -1 \quad x_2 = 1 \quad t = 1$

$w_1 = 0 \quad w_2 = -2 \quad b = 0$

$y_{in} = (-1\times0) + (1\times-2) + 0$

$y_{in} = -2 \quad y_{out} = -1$

$t = 1 \neq y_{out} = -1$

$w_1(n) = 0 + 1\times1\times-1$

$w_1(n) = -1$

$w_2(n) = -2 + 1\times1\times1$

$w_2(n) = -1$

$b(n) = 0 + 1\times1$

$b(n) = +1$

$y_{in} = 1 + (-1) + +1$

$y_{in} = +1 \quad y_{out} = 1$

(target = 1) ≠ (yout = -1) (target = 1) == (yout = 1)

update the weights

$w_{1(n)} = w_{0(n)} + \alpha t x_1$

$w_{1(n)} = 1 + 1 \times 1 \times -1$

$w_{1(n)} = 0$

$w_{2(n)} = w_{2(0)} + \alpha t x_2$

$w_{2(n)} = 0$

$b_{(n)} = b_0 + \alpha t$

$b = 2$

(4) $x_1 = -1$ $x_2 = -1$ $x_1 = -1$ $x_2 = 1$ $t = 1$
$t = 1$
$w_1 = -1$ $w_2 = -1$ $b = 1$ $w_1 = 0$ $w_2 = 0$ $b = 2$

$y_{in} = 1 + 1 + 1$

$y_{in} = 3$

$y_{out} = 1$

(t = 1) == (yout = 1)

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

$y_{in} = 2$ $y_{out} = f(y_{in})$

(target = 1) == (yout = 1)

(4) For 4th set of inputs

$x_1 = -1$ $x_2 = -1$ $t = 1$

$w_1 = 0$ $w_2 = 0$ $b = 2$

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

$y_{in} = (-1 \times 0) + (-1 \times 0) + 2$

$y_{in} = 2$

$y_{out} = f(y_{in}) = f(2) = 1$

(target = 1) == (yout = 1)

x_1	x_2	$t = x_1$ NAND x_2	yout
1	1	-1	-1
1	-1	1	1
-1	1	1	1
-1	-1	1	1

$w_1 = -1$ $w_2 = -1$ $b = 1$

$x_1 w_1 + x_2 w_2 + b$

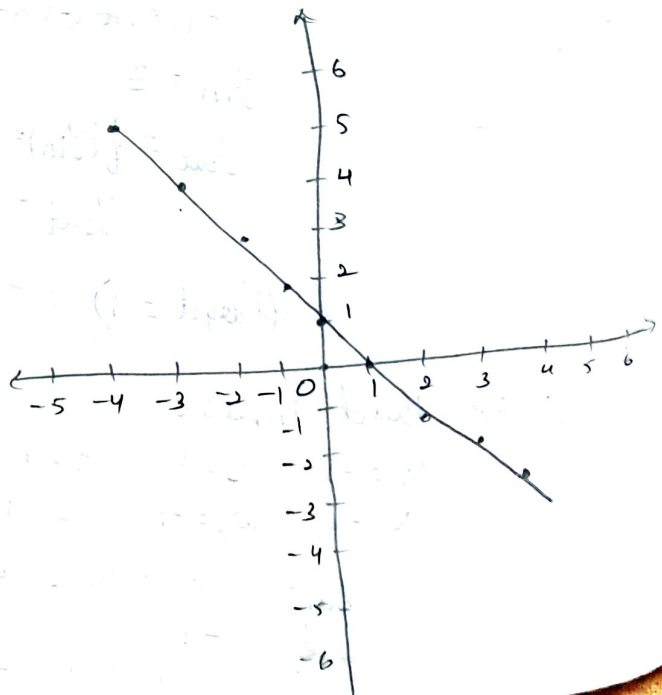
$= x_1(-1) + x_2(-1) + 1$

$= -x_1 - x_2 + 1$

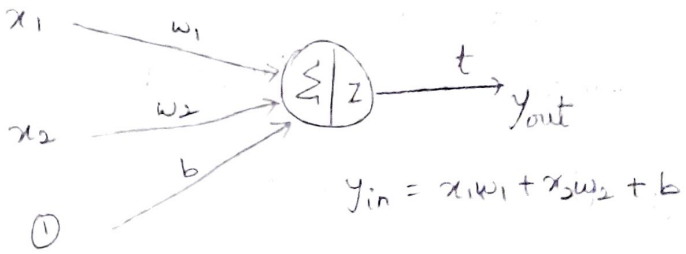
$x_2 = 1 - x_1$

- For unseen data

x_1	$x_2 = 1 - x_1$
-4	1 - (-4) = 5
-3	4
-2	3
-1	2
0	+1
1	0
2	-1
3	-2
4	-3



(iii) **NOR**



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

Given

$w_1 = w_2 = b = 0$, $\alpha = 1$ $\theta = 1$

x_1	x_2	$t = x_1$ NOR x_2	$y_{out} = ?$
1	1	-1	?
1	-1	-1	
-1	1	-1	
-1	-1	1	

Activation function - $y_{out} = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 1 \\ -1 & \text{if } y_{in} < 1 \end{cases}$

① For first set of inputs

$x_1 = 1$ $x_2 = 1$ $t = -1$

$w_1 = 0$ $w_2 = 0$ $b = 0$

$$y_{in} = x_1 w_1 + x_2 w_2 + b$$
$$= 1 \times 0 + 1 \times 0 + 0$$
$$y_{in} = 0$$
$$y_{out} = f(y_{in}) = f(0)$$
$$y_{out} = -1$$

$$\boxed{(target = -1) == (y_{out} = -1)}$$

② For second set of inputs

$x_1 = 1$ $x_2 = -1$ $t = -1$

$w_1 = 0$ $w_2 = 0$ $b = 0$

$$y_{in} = x_1 w_1 + x_2 w_2 + b$$
$$= 1 \times 0 + (-1) \times 0 + 0$$
$$y_{in} = 0$$

$$y_{out} = f(y_{in}) = f(0)$$

$$y_{out} = -1$$

$$\boxed{(target == -1) \quad == \quad (y_{out} = -1)}$$

③ for third set of inputs

$x_1 = -1 \qquad x_2 = 1 \qquad t = -1$

$w_1 = 0 \qquad w_2 = 0 \qquad b = 0$

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

$$y_{in} = (-1 \times 0) + (1 \times 0) + 0$$

$$y_{in} = 0$$

$$y_{out} = f(y_{in}) = f(y_0)$$

$$y_{out} = -1$$

$$\boxed{(target = -1) \quad == \quad (y_{out} = -1)}$$

④ For fourth set of inputs

$x_1 = -1 \qquad x_2 = -1 \qquad t = 1$

$w_1 = 0 \qquad w_2 = 0 \qquad b = 0$

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

$$y_{in} = (-1) \times 0) + (-1 \times 0) + 0$$

$$y_{in} = 0$$

$$y_{out} = f(y_{in}) = f(0)$$

$$y_{out} = -1$$

$$\boxed{target = 1 \quad \neq \quad y_{out} = -1}$$

→ update the weights

$$w_{1(n)} = w_{1(0)} + \alpha t x_1$$

$$w_{1(n)} = 0 + 1 \times 1 \times -1$$

$$w_{1(n)} = -1$$

$$w_{2(n)} = w_{2(0)} + \alpha t x_2$$

$$w_{2(n)} = 0 + 1 \times 1 \times -1$$

$$w_{2(n)} = -1$$

$$b_{(n)} = b_{(0)} + \alpha t$$

$$b_{(n)} = 0 + 1 \times 1$$

$$b = 1$$

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

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

$$y_{in} = 3$$

$$y_{out} = f(y_{in}) = f(3)$$

$$y_{out} = 1$$

$$\text{target} = 1 \quad == \quad y_{out} = 1$$

→ For the lineal class

$$w_1 x_1 + w_2 x_2 + b = 0$$

$$w_1 = -1 \quad w_2 = -1 \quad b = 1$$

$$-x_1 - x_2 + 1 = 0$$

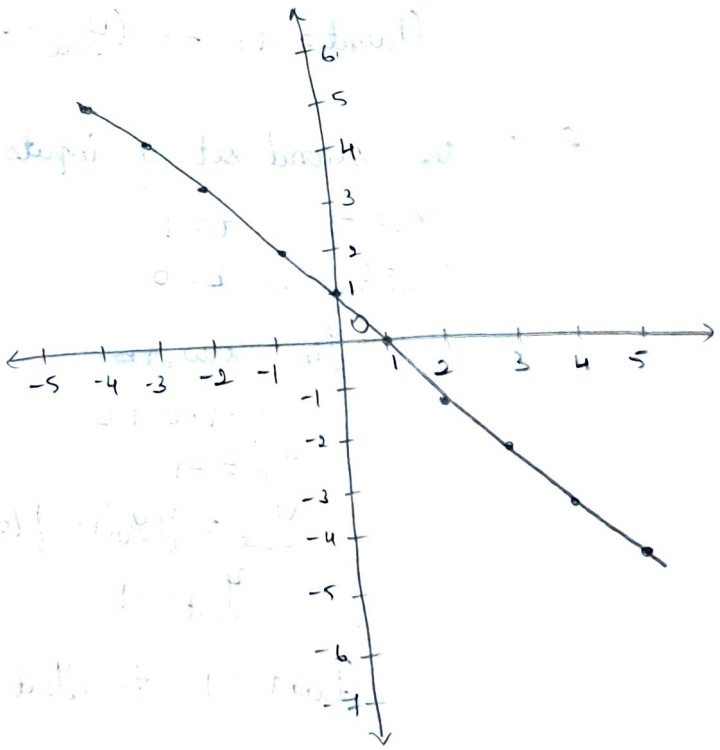
$$x_1 + x_2 = 1$$

$$x_1 = 1 - x_2$$

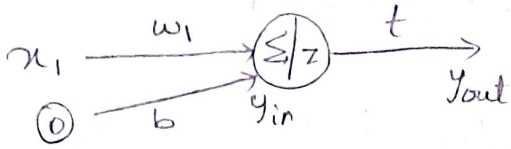
x_1	x_2	$t = x_1$ NOR x_2	y_{out}
1	1	-1	-1
1	-1	-1	-1
-1	1	-1	-1
-1	-1	1	1

→ For unseen data

x_1	$x_2 = 1 - x_1$
-4	$1-(-4) = 5$
-3	4
-2	3
-1	2
0	1
1	0
2	-1
3	-2
4	-3



(iv) NOT



$$y_{in} = x_1 w_1 + b$$

Given $w_1 = w_2 = b = 0$, $\alpha = 1$, $\theta = 1$

x_1	t = NOT x_1	y_{out} = ?
1	-1	?
-1	1	

Activation function $y_{out} = f(y_{in})$

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

① For the first set of inputs

$x_1 = 1$ $t = -1$

$w_1 = 0$ $b = 0$

$y_{in} = x_1 w_1 + b$

$= 1 \times 0 + 0$

$y_{in} = 0$

$y_{out} = f(y_{in}) = f(0)$

$y_{out} = -1$

(target = -1) == (y_{out} = -1)

② For the second set of inputs

$x_1 = -1$ $t = 1$

$w_1 = 0$ $b = 0$

$y_{in} = x_1 w_1 + b$

$= -1 \times 0 + 0$

$y_{in} = -1$

$y_{out} = f(y_{in}) = f(0)$

$y_{out} = -1$

(target = 1) $\neq$ (y_{out} = -1)

→ update the weights

$$w_1(n) = w_{(0)} + \alpha t x_1$$
$$= 0 + 1 \times 1 \times -1$$
$$w_{1(n)} = -1$$

$$b_{(n)} = b_{(0)} + \alpha t$$
$$b_{(n)} = 0 + 1 \times 1$$
$$b_{(n)} = 1$$

$x_1 = -1 \quad t = 1$

$w_1 = -1 \quad b = 1$

$$y_{in} = x_1 w_1 + b$$
$$= -1 \times -1 + 1$$
$$y_{in} = 2$$
$$y_{out} = f(y_{in}) = f(2)$$
$$y_{out} = 1$$

(target = 1) == (y_{out} == 1)

x_1	t = NOT(x_1)	y_{out}
1	-1	-1
-1	1	1

ASSIGNMENT - 3

Basics Of ANN

1) Write the purpose of weights in the artificial neural network models.

Weights in artificial neural network (ANN) models play a crucial role in determining the network's functionality and learning ability. The purpose includes

(i) Capturing Relationships between inputs and outputs weights represent the strength and direction of the connection between neurons. They determine how many influence an input feature or previous layer's output has on the subsequent layer.

(ii) Facilitating Learning :

During training, weights are adjusted through backpropagation and optimization algorithms (e.g gradient descent). This adjustment enables the network to minimize the loss function and improve its predictive accuracy.

(iii) Storing Knowledge :

The learned weights encode patterns and information about the training data. These weights collectively define the model's ability to generalize to unseen data.

(iv) Enabling Feature Scaling :

Weights scale the input data, enabling the network to continue & modify input features in ways that are optimal for a given task, such as classification or regression.

(v) Providing Flexibility :

By adjusting weights, ANNs can adapt to different problems and datasets, making them versatile for various applications such as image recognition, natural language processing, and time series prediction.

(vi) Supporting Non-linearity :

Combined with activation functions, weights help the network model complex, non-linear relationships in data, which is crucial for solving real-world problems.

(vii) Contributing to Decision Boundaries :

Weights help define the decision boundaries in classification tasks, enabling the network to distinguish between different classes in the input data.

2) Illustrate the uses of different activation functions for artificial neural networks.

Activation functions are mathematical functions applied to the output of a neuron in an artificial neural network (ANN) to introduce non-linearity and enable the network to learn complex patterns.

Different activation functions have specific characteristics that make them suitable for various tasks.

(i) Sigmoid Activation Function :

Formula : $\sigma(x) = \frac{1}{1+e^{-x}}$

Range : $(0, 1)$

Uses : Ideal for binary classification tasks, as it maps outputs to a probability range $(0, 1)$.

- Suitable for the output layer of binary logistic regression models

Limitations: Vanishing gradient problem for large positive & negative inputs. Outputs are not zero-centered.

(ii) Hyperbolic Tangent (Tanh) Activation Functions:

Formula: $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

Range: $(-1, 1)$

Uses: often used in hidden layers for networks requiring zero-centered outputs

: Suitable for modeling data where negative outputs are meaningful.

Limitations - suffers from the vanishing gradient problem, similar to the sigmoid function

(iii) Rectified Linear Unit (ReLU) Activation Function:

Formula: $f(x) = \max(0, x)$

Range: $[0, \infty]$

Uses: widely used in hidden layers of deep neural networks due to its simplicity and computational efficiency. Mitigates the vanishing gradient problem by maintaining gradients for positive inputs.

Limitations: Can suffer from the "dying ReLU" problem, where neurons become inactive for all input.

(iv) Leaky ReLU Activation Function:

Formula: $f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{otherwise} \end{cases}$

where α is a small positive value (eg, 0.01)

Range : $(-\infty, \infty)$

Uses : Addresses the "dying ReLU" problem by allowing small gradients for negative inputs. Suitable for deep networks with sparse data.

(v) Softmax Activation Function :

Formula : $\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1} e^{x_j}}$

Range : (0,1), summing to 1 across all outputs

Uses : Commonly used in the output layer for multi-class classification tasks. Converts raw scores into class probabilities.

(vi) Linear Activation Function :

Formula : $f(x) = x$

Range : $(-\infty, \infty)$

Uses : used in the output layer for regression tasks suitable when the output is continuous.

Limitations : Cannot introduce non-linearity, making it unsuitable for hidden layers.

(vii) Exponential Linear Unit (ELU)

Formula : $f(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha(e^x - 1) & \text{otherwise} \end{cases}$

where $\alpha > 0$

Range : $(-\alpha, \infty)$

Uses : Addresses the dying neuron problem by smoothing the negative part of the output.

Retains benefits of ReLU, while allowing negative values.

(viii) Swish Activation Function

Formula: $f(x) = x \cdot \sigma(x)$,
where $\sigma(x)$ is the sigmoid function

Range: $(-\infty, \infty)$

Uses: works well in deep network and provides smoth gradients
often outperforms ReLU in specific architecture.

3) Write the algorithm used in a single layer perceptron model for learning.

The single layer perceptron (SLP) is a type of neural network that uses a single layer of weights to map input features to output. its learning process involves adjusting weights to minimize errors in classification or regression. Below is the step-by-step algorithm used for training a single-layer perceptron.

Single Layer perceptron Learning Algorithm:

① Initialize Parameters:
- Randomly initialize the weight vector "w" and bias "b" with small random values (eg. close to zero)
- Set the learning rate η (a small positive constant)

② Input and Output:
- prepare the training dataset with 'N' samples, where each choose sample consists of:
  - Input feature vector $x_i = [x_{i1}, x_{i2}, \ldots, x_{it}]$ (of size t)
  - Target output y_i (e.g., 0 or 1 for binary classification)

③ Activation Function :

• Use a step function (& other suitable activation functions) to compute the perceptron's output :

$$\hat{y}_i = f(w_i x_i + b) \quad \text{where } f(z) \text{ is typically}$$

$$f(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

④ Training Loop :

• For each training epoch (iteration over the dataset)

• For Each training sample (x_i, y_i) : (i) Calculate the predicted output

$$\hat{y}_i = f(w \cdot x_i + b)$$

(ii) Compile the errors :

$$e_i = y_i - \hat{y}_i$$

(iii) update the weights and bias :

if $e_i \neq 0$ (i.e there is an error)

$$w = w + \eta e_i x_i$$

$$b = b + \eta e_i$$

⑤ Stopping criteria :

• Continue the training loop until

• Errors are minimized (e.g no misclassified samples)

• A maximum number of epochs is reached.

⑥ Output the Final Model :

• The perceptron learns the final weight vector 'w' and bias 'b', which defines the decision boundary.

Key points : The learning process adjusts weights iteratively to reduce the classification error.

A single layer perceptron can only solve linearly seperable problems.

If the data is not linearly seperable, the perceptron will fail to converge.

1.) Distinguish the following

(a) threshold

(b) Learning Rate

(c) Activation

(a) Threshold

- Definition - The threshold is a parameter that determines wheather the output of a neuron is activated (fires) or not, based on the weighted sum of inputs.
- It is used in conjunction with a step function or similar activation functions.

Purpose :

To define a boundary or limit for the neuron to produce an output.

In simple perceptrons, if the weighted sum of inputs exceeds the threshold, the output is 1, otherwise it is 0.

Role in a Model :

Threshold often appears as a bias term 'b', which is learned during training to shift the decision boundary For instance, the perceptron's output is determined by

$$\text{output} = f(wx + b)$$

(b) Learning Rate (η)

- **Definition :**

The learning rate is a hyperparameter that controls the step size during weight updates in the training process.

- **Purpose :**

To regulate how much the weights are adjusted in response to the error for each iteration.

Ensures smoth convergence to an optimal solution during training.

Role in the model :

Used in weight and bias update rules

$$w = w + \eta e x$$
$$b = b + \eta e$$

here, η is the learning rate, and e is the error.

Key considerations :

- A small learning rate ensures slow but stable convergence reducing the risk of overshooting the optimal solution.
- A large learning rate may speed up convergence but can lead to oscillations or devergence.

(c) Activation

- **Definition :**

- Activation refers to the function applied to the weighted sum of inputs (including bias) to determine the output of a neuron.

- Common activation functions include step, sigmoid, ReLU and softmax.

• Purpose :

- To introduce non-linearity in to the model, enabling it to learn complex patterns and relationships

- Determines the output of a neuron based on the weighted sum and bias.

• Role in a Model :

Transforms the input signal into an output signal based on the chosen function.

$$\text{Output} = f(wx + b)$$

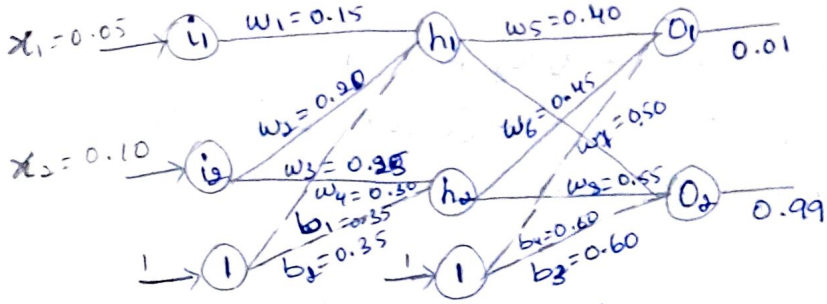
$f(z)$ could be linear, non-linear or threshold-based.

Linear - Directly passes the input (used in regression)

Non Linear - Includes sigmoid, ReLU, tanh, etc which enables the model to solve complex, non-linear problems

ASSIGNMENT - 4

1) Implement Backpropagation model algorithm.



For a single training set inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99.

Use backpropagation algorithm to show the updated weights after the first forward pass.

Solution:

Steps

(i) Feed forward for first layer
- Calculate net inputs / Activation = output

(ii) Feed forward for second layer
- Calculate net inputs / Activation = output

(iii) Error back propagation

(iv) update the weights.

$[w_1, w_2, w_3, w_4] = [0.15 \quad 0.20 \quad 0.25 \quad 0.30]$

$[w_5, w_6, w_7, w_8] = [0.40 \quad 0.45 \quad 0.50 \quad 0.55]$

$[b_1, b_2] = [0.35, \ 0.35]$

$[b_3, b_4] = [0.60, \ 0.60]$

Assuming $\alpha = 0.5$ $\qquad [x_1, x_2] = [0.05 \quad 0.10]$

(i) Feed Forward

→ Net inputs at first/hidden/output layer

hin_1 → Net input at h_1 → hin_1

$$hin_1 = x_1 w_1 + x_2 w_3 + b_1 \times 1$$

$$hin_1 = (0.05 \times 0.15) + (0.10 \times 0.25) + (0.35 \times 1)$$

$$\boxed{hin_1 = 0.3825}$$

$$hin_2 = x_2 w_4 + (x_1 * w_2) + b_2 \times 1$$

$$hin_2 = (0.10 \times 0.30) + (0.05 \times 0.20) + 0.35$$

$$\boxed{hin_2 = 0.39}$$

→ Activation function at input layer h

$$h_1 = f(hin_1) = \frac{1}{1+e^{-(0.3825)}} = 0.59447$$

$$h_2 = f(hin_2) = \frac{1}{1+e^{-(0.39)}} = 0.59628$$

$$\boxed{h_{1out} = 0.59447}$$

$$\boxed{h_{2out} = 0.59628}$$

(ii) Net inputs at second layer (O)

$$O_{in_1} = h_{1out} \times w_5 + h_{2out} \times w_7 + b_3 \times 1$$

$$O_{in_1} = (0.59447 \times 0.4) + (0.59628 \times 0.5) + 0.6$$

$$\boxed{O_{in_1} = 1.1359}$$

$$O_{in_2} = h_{2out} \times w_8 + h_{1out} \times w_6 + b_4 \times 1$$

$$O_{in_2} = (0.59628 \times 0.55) + (0.59447 \times 0.45) + 0.6$$

$$\boxed{O_{in_2} = 1.19546}$$

→ Activation function at layer (O)

$$O_{1out} = f(O_{in1}) = \frac{1}{1+e^{-(1.1359)}} = 0.75692$$

$$O_{2out} = f(O_{in2}) = \frac{1}{1+e^{-(0.19546)}} = 0.76755$$

(iii) Total Error at the output layer O

→ Total Error $= E = \frac{1}{2}(T_1 - Out_1)^2 + \frac{1}{2}(T_2 - O_{out2})^2$

$$E = 0.5(0.01 - 0.75692)^2 + \frac{1}{2}(0.99 - 0.76755)^2$$

$$\boxed{E = 0.30368}$$

Error at δO_1

$$\delta O_1 = (T_1 - O_{out1}) \times O_{out1} * (1 - O_{out1})$$

$$\delta O_1 = (0.01 - 0.75692) \times 0.75692 \times (1 - 0.75692)$$

$$\boxed{\delta O_1 = -0.137427}$$

Error at δO_2

$$\delta O_2 = (T_2 - O_{out2}) * O_{out2} * (1 - O_{out2})$$

$$\delta O_2 = (0.99 - 0.76755) * 0.76755 * (1 - 0.76755)$$

$$\boxed{\delta O_2 = 0.039688}$$

→ Update the weights (Backpropagation)

$$w_5(n) = w_5(0) + \alpha \delta O_1 * h_{1out}$$

$$= 0.4 + 0.5 \times (-0.13742) * 0.59447$$

$$w_5(n) = 0.35915$$

$$w_6(n) = w_6(0) + \alpha \delta O_1 * h_{2out}$$

$$= 0.45 + (0.5 * (-0.13742)) * 0.59628)$$

$$w_6(n) = 0.40902$$

$$b_2(n) = b_3(0) + \alpha \delta O_1 * 1$$

$$b_3(n) = 0.6 + (0.5 * (-0.13742))$$

$$b_3(n) = 0.53109$$

$$w_{7(n)} = w_{7(0)} + \alpha \delta o_2 * h_{1out}$$

$$w_{7(n)} = 0.5 + (0.5 * 0.03968 * 0.59477)$$

$$w_{7(n)} = 0.51180$$

$$w_{8(n)} = w_{8(0)} + \alpha \delta o_2 * h_{2out}$$

$$= 0.55 + 0.5 \times (0.03968) * 0.59628$$

$$w_{8(n)} = 0.56183$$

$$b_{4(n)} = b_{4(0)} + \alpha \delta o_2 * 1$$

$$b_{4(n)} = 0.6 + (0.5 \times 0.03963)$$

$$b_{4(n)} = 0.61984$$

→ Calculating Error at layer h

$$\delta h_1 = (\delta o_1 * w_5 + \delta o_2 * w_6) * (h_{1out} * 1 - h_{1out}))$$

$$\delta h_1 = ((-0.137427 * 0.40) + (0.03968 * 0.45)) * (0.59447 * (1 - 0.59447))$$

$$\boxed{\delta h_1 = -0.008947}$$

$$\delta h_2 = (\delta o_1 * w_7 + \delta o_2 * w_8) * h_{2out} * (1 - h_{2out})$$

$$\delta h_2 = ((-0.137427 * 0.5) + 0.03963 * 0.55)) * (0.59628 * (1 - 0.5962))$$

$$\boxed{\delta h_2 = -0.011287}$$

→ weight calculation at h

$$w_{1(n)} = w_{1(0)} + \alpha \delta h_1 * i_1$$

$$w_{1(n)} = 0.15 + (0.5 * (-0.008947) * 0.05)$$

$$w_{1(n)} = 0.14977$$

$$w_{2(n)} = w_{2(0)} + \alpha \delta h_2 * i_1$$

$$w_{2(n)} = 0.20 + ((0.5 * (-0.011287) * 0.05)$$

$$w_{2(n)} = 0.19997$$

$$w_{3(n)} = w_{3(0)} + \alpha \delta h_1 * i_2$$

$$w_{3(n)} = 0.25 + ((0.5 * -0.008947) * 0.10)$$

$$w_{3(n)} = 0.24955$$

$$w_4(n) = w_4(0) + \alpha \delta h_2 * i_2$$

$$w_4(n) = 0.3 + (0.5 * (-0.011287) * 0.10)$$

$$w_4(n) = 0.29994$$

$$b_1(n) = b_1(0) + \alpha \delta h_1 * 1$$

$$b_1(n) = 0.35 + 0.5 * (-0.008947) * 1$$

$$b_1(n) = 0.34552$$

$$b_2(n) = b_2(0) + \alpha \delta h_2 * 1$$

$$b_2(n) = 0.35 + 0.5 \times (-0.011287) * 1$$

$$b_2(n) = 0.34435$$

All the new updated weights are replaced with old weights.

$w_1(0) = w_1(n)$ $\quad w_2(0) = w_2(n)$ $\quad w_3(0) = w_3(n)$ $\quad b_1(0) = b_1(n)$

$w_4(0) = w_4(n)$ $\quad w_5(0) = w_5(n)$ $\quad w_6(0) = w_6(n)$ $\quad b_2(0) = b_2(n)$

$w_7(0) = w_7(n)$ $\quad w_8(0) = w_8(n)$ $\quad b_3(0) = b_3(n)$ $\quad b_4(0) + b_4(n)$

End of Epoch 1.

Submitted by :- Sufiya Tasanum

Roll No : 24105038

Big Data Analytics