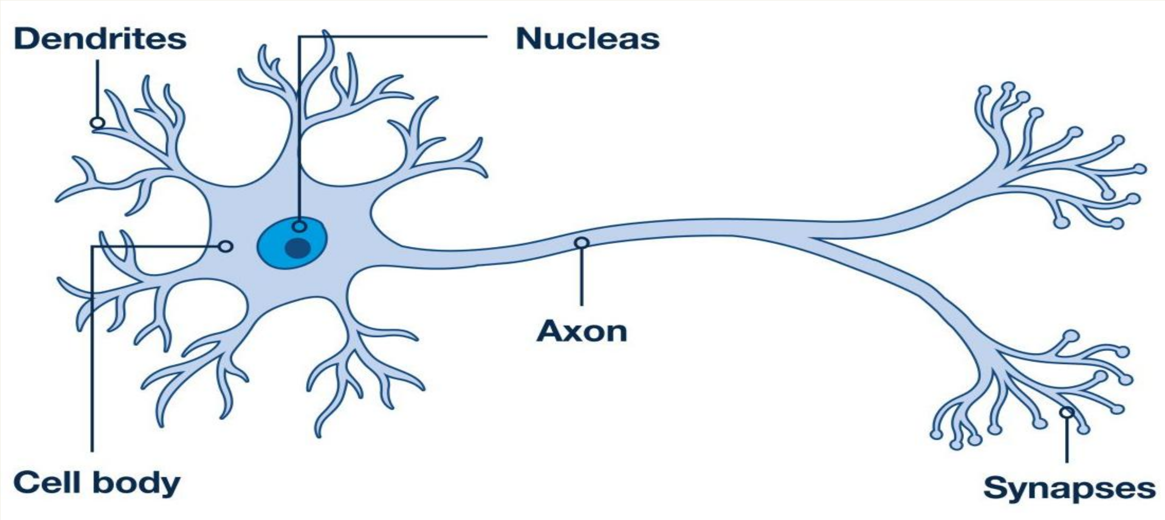
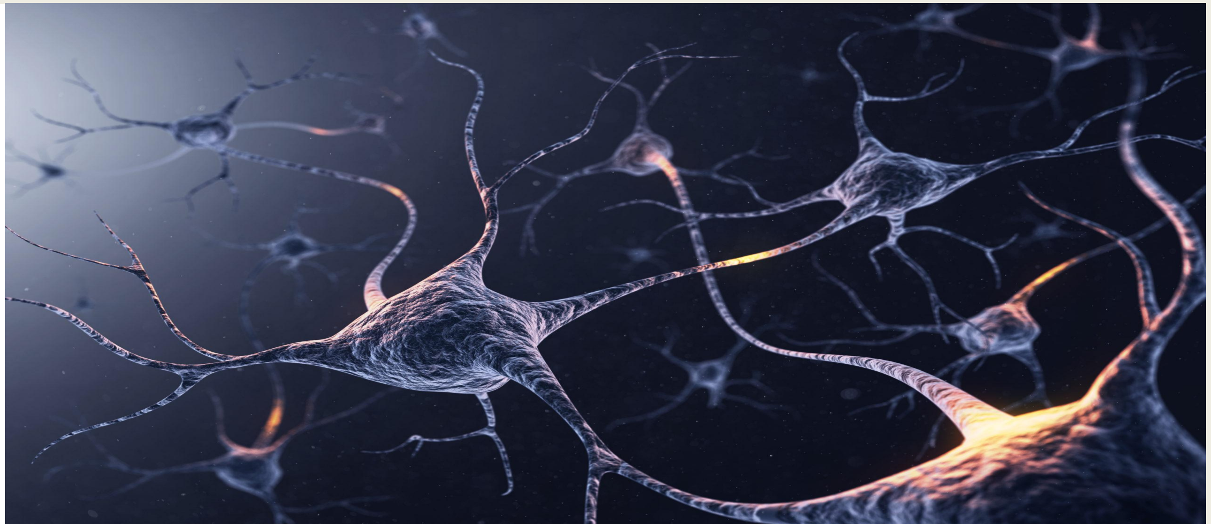
ML
LECTURE-21

BY
Dr. Ramesh Kumar Thakur
Assistant Professor (II)
School Of Computer Engineering

Neural networks

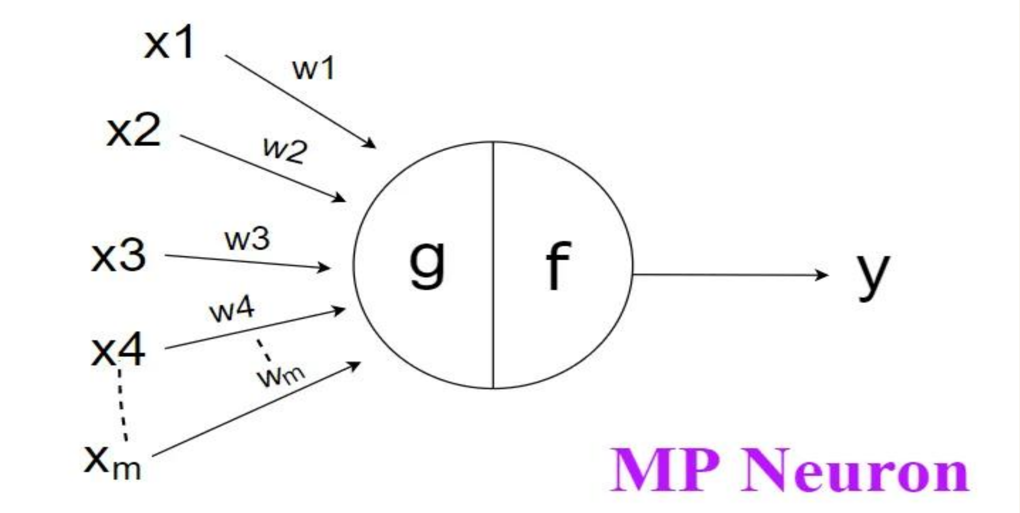
- An **Artificial Neural Network (ANN) models the relationship between a set of input signals and an output signal** using a model derived from our **understanding of how a biological brain responds to stimuli from sensory inputs.**
- Just as a brain uses a network of interconnected cells called neurons to create a massive parallel processor, **ANN uses a network of artificial neurons or nodes to solve learning problems.**
- **<u>Biological motivation</u>**
- In the cell, the **incoming signals are received by the cell's dendrites** through a biochemical process.
- The process allows the **impulse to be weighted according to its relative importance or frequency.**
- As the **cell body begins accumulating the incoming signals, a threshold is reached at which the cell fires and the output signal is transmitted via an electrochemical process down the axon.**
- At the axon's terminals, the **electric signal is again processed as a chemical signal to be passed to the neighboring neurons across a tiny gap known as a synapse.**





McCulloch - Pitts Neuron (MP Neuron)

- The MP neuron is **mankind's first simplified mathematical model of the neuron.**
- This model was **developed by McCulloch and Pitts in 1943.**
- The MP neuron model is also known as the **linear threshold gate model.**
- They are widely used in proving logic functions.



- Now let's look into the model. It has **4 basic components :**
- The model **takes inputs (x1, x2, …., xm)**
- **Applies Adder function(g)** and
- Takes **decision in Activation function (f)**
- **Gives an output Y**

Characteristics of MP Neuron

- When MP neurons are modeled as neural networks, they are connected by directed weighted paths in a neural network.
- When we pass the Adder function value in the Activation function of an MP neuron, there are 2 possibilities: the neuron may fire (label 1) or it does not fire (label 0).
- The activation function is based on the threshold value.
- There is a **fixed threshold for each neuron and if the net input to the neuron is greater than the threshold then the neuron fires.**
- **Weights(w):** It is the parameter that shows the **contributing power of the input feature towards the output.**
- Low weight value will have no change on the input and high weight will have a more significant change on the output.
- **Adder Function(g):** It is an Aggregation function that performs the **sum of the product of the inputs with the weights** and gets Adder value.
- **Activation Function(f):** It is the **mathematical function** that **decides whether neuron input is relevant for model prediction or not.**
- **Threshold value(b):** It is the value in the activation function based on which the activation function takes its decision.
- Initially, we take any value from 0 to n (where n is maximum adder value) and then iterate over it and find the total loss of the model.
- Then we can **choose the value of the threshold, such that the loss is minimum.** This is the brute force method by which we calculate the threshold value.

Characteristics of MP Neuron

$$g(x) = \sum_{i=1}^{m} w_i x_i$$

$$y = f(g(x)) = 1 \text{ if } g(x) \geq b$$
$$= 0 \text{ if } g(x) < b$$

- From the above image :
- **g(x) is the aggregated sum of all weighted inputs**
- **y is the final value predicted by an activation function (f).**
- **b is the threshold value which is calculated by the brute force method**

AND function using MP Neuron

- Consider the truth table for AND function
- The M–P neuron has no particular training algorithm
- In M-Pneuron, only analysis is being performed.
- Hence, assume the weights be w1 = 1 and w2 = 1.

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

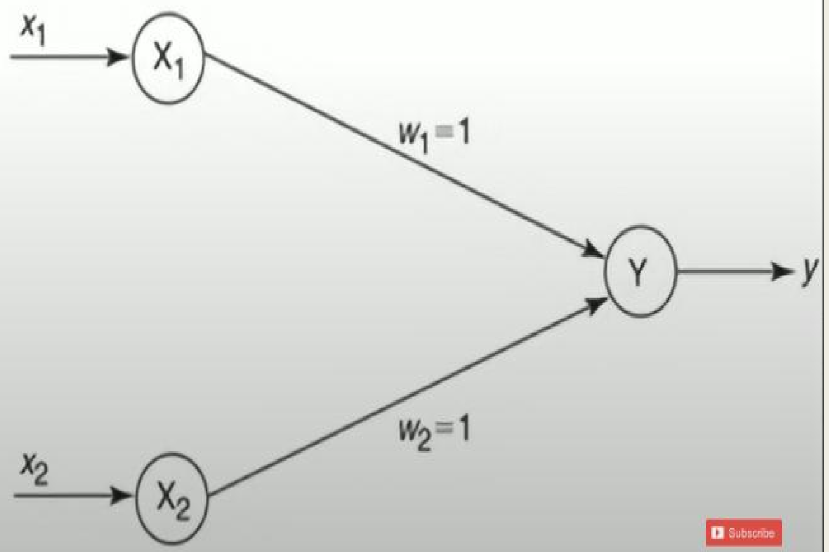
$$(1, 1),\ y_{in} = x_1 w_1 + x_2 w_2 = 1 \times 1 + 1 \times 1 = 2$$

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

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

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

Threshold value is set equal to 2 ($\theta = 2$).



ANDNOT function using MP Neuron

- Consider the truth table for ANDNOT function
- The M–P neuron has no particular training algorithm
- In M-P neuron, only analysis is being performed.
- Hence, assume the weights be w1 = 1 and w2 = 1.

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

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

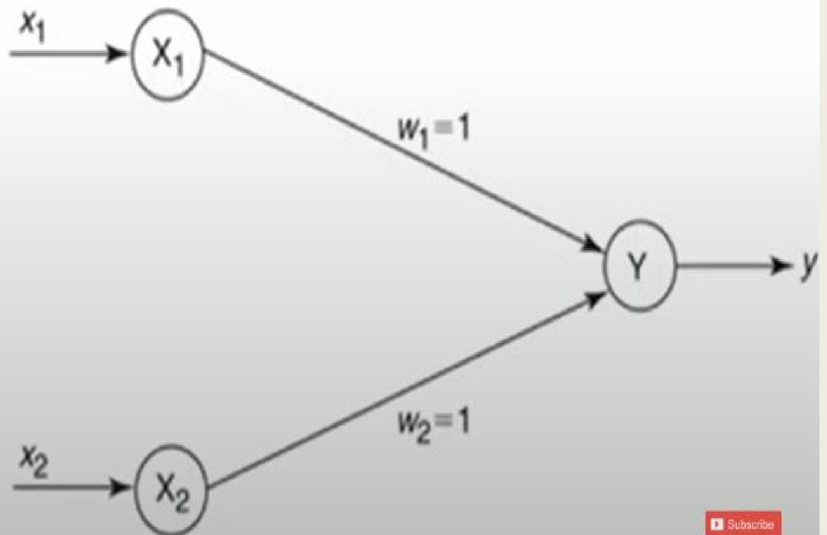
$(1, 1),\ y_{in} = 1 \times 1 + 1 \times 1 = 2$

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

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

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

From the calculated net inputs, it is not possible to fire the neuron for input (1, 0) only. Hence, these weights are not suitable.



ANDNOT function using MP Neuron

- Consider the truth table for ANDNOT function
- The M–P neuron has no particular training algorithm
- In M-P neuron, only analysis is being performed.
- Hence, assume the weights be w1 = 1 and w2 = -1.

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

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

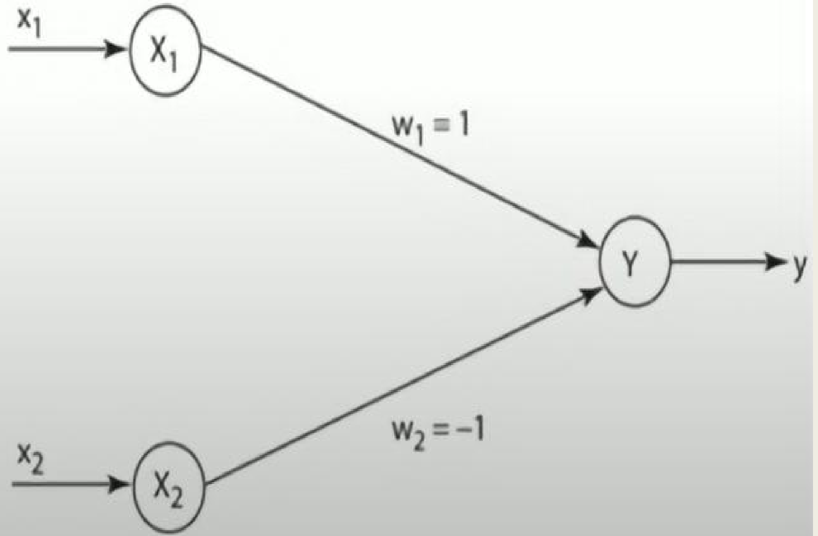
$(1, 1),\ y_{in} = 1 \times 1 + 1 \times -1 = 0$

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

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

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

From the calculated net inputs, now it is possible to fire the neuron for input (1, 0) only by fixing a threshold of 1, i.e., $\theta \geq 1$ for Y unit.



Limitations and Solution of M-P Neuron

- **Limitations of MP Neuron**
- What about non-boolean (say, real) inputs?
- Do we always need to hand code the threshold?
- Are all inputs equal? What if we want to assign more importance to some inputs?
- What about functions which are not linearly separable? Say XOR function.

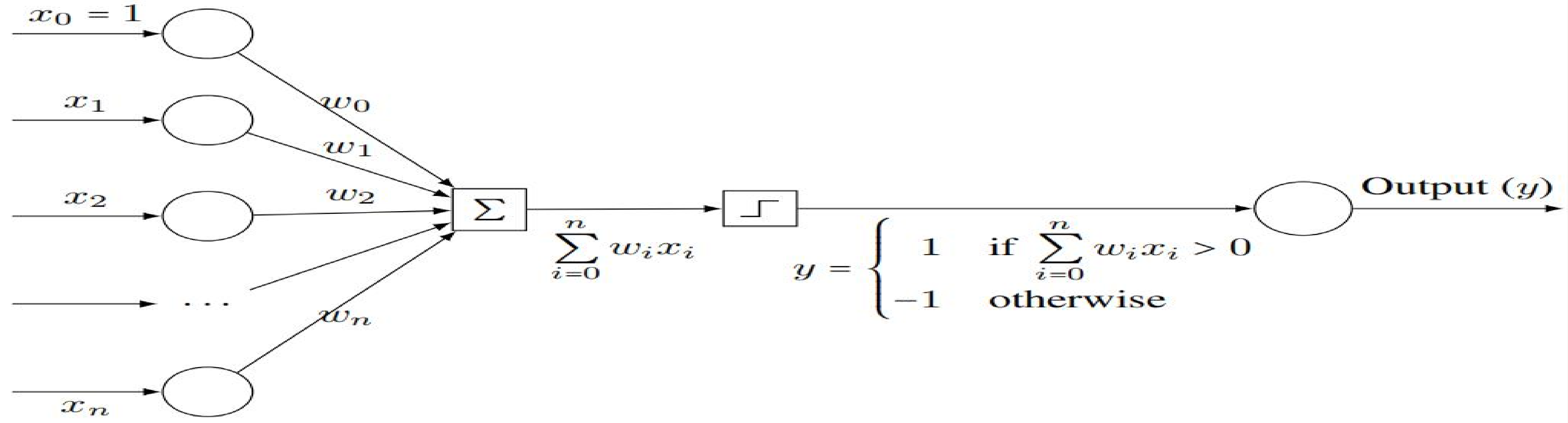
- **Solution for limitations of MP Neuron**
- Overcoming the limitations of the M-P neuron, **Frank Rosenblatt**, an American psychologist, proposed the **classical perception model, the mighty artificial neuron (perceptron), in 1957.**
- It is **more generalized computational model than the McCulloch-Pitts neuron where weights and thresholds can be learnt over time.**

Perceptron Model

- A perceptron is an artificial neuron in which the **activation function is the threshold function.**
- Consider an artificial neuron having x1, x2, ⋯ , xn as the input signals and w1, w2, ⋯ , wn as the associated weights.
- Let w0 be some constant (known as **bias**).
- The neuron is called a perceptron if the output of the neuron is given by the following function:

$$o(x_1, x_2, \ldots, x_n) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + \cdots + w_n x_n > 0 \\ -1 & \text{if } w_0 + w_1 x_1 + \cdots + w_n x_n \leq 0 \end{cases}$$

- Below figure shows the schematic representation of a perceptron.



Perceptron Training Rule

```
Perceptron_training_rule (X, η)
initialize w (wi ← an initial (small) random value)
repeat
    for each training instance (x, tx) ∈ X
        compute the real output ox = Activation(Summation(w.x))
        if (tx ≠ ox)
            for each wi
                wi ← wi + Δwi
                Δwi ← η (tx - ox)xi
            end for
        end if
    end for
until all the training instances in X are correctly classified
return w
```

AND gate using Perceptron Model

w1 = 1.2, w2 = 0.6 Threshold = 1 and Learning Rate n = 0.5

A	B	A^B
0	0	0
0	1	0
1	0	0
1	1	1

1. A=0, B=0 and Target = 0
   - *wi.xi = 0*1.2 + 0*0.6 = 0*
   - This is not greater than the threshold of 1, so the output = 0
2. A=0, B=1 and Target = 0
   - *wi.xi = 0*1.2 + 1*0.6 = 0.6*
   - This is not greater than the threshold of 1, so the output = 0

AND gate using Perceptron Model

w1 = 1.2, w2 = 0.6 Threshold = 1 and Learning Rate n = 0.5

A	B	A^B
0	0	0
0	1	0
1	0	0
1	1	1

3. A=1, B=0 and Target = 0

- $wi.xi = 1*1.2 + 0*0.6 = 1.2$
- This is greater than the threshold of 1, so the output = 1

$$wi = wi + n(t - o)xi$$

$$w1 = 1.2 + 0.5(0 - 1)1 = 0.7$$

$$w2 = 0.6 + 0.5(0 - 1)0 = 0.6$$

AND gate using Perceptron Model

w1 = 0.7, w2 = 0.6 Threshold = 1 and Learning Rate n = 0.5

A	B	A ^ B
0	0	0
0	1	0
1	0	0
1	1	1

1. A=0, B=0 and Target = 0
   - *wi.xi = 0*0.7 + 0*0.6 = 0*
   - This is not greater than the threshold of 1, so the output = 0
2. A=0, B=1 and Target = 0
   - *wi.xi = 0*0.7 + 1*0.6 = 0.6*
   - This is not greater than the threshold of 1, so the output = 0

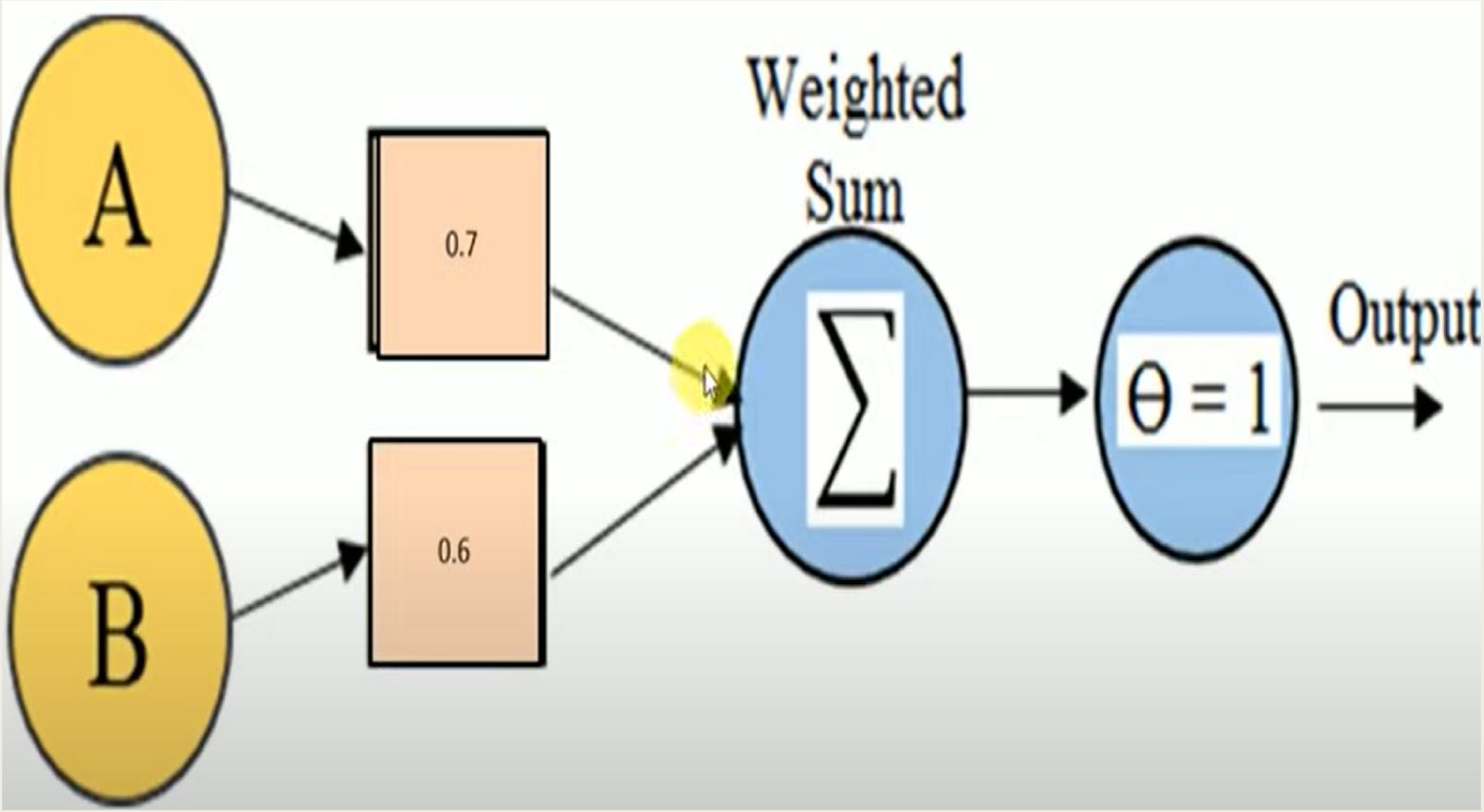
AND gate using Perceptron Model

w1 = 0.7, w2 = 0.6 Threshold = 1 and Learning Rate n = 0.5

A	B	A ^ B
0	0	0
0	1	0
1	0	0
1	1	1

3. A=1, B=0 and Target = 0
   - *wi.xi = 1*0.7 + 0*0.6 = 0.7*
   - This is not greater than the threshold of 1, so the output = 0
4. A=1, B=1 and Target = 1
   - *wi.xi = 1*0.7 + 1*0.6 = 1.3*
   - This is greater than the threshold of 1, so the output = 1

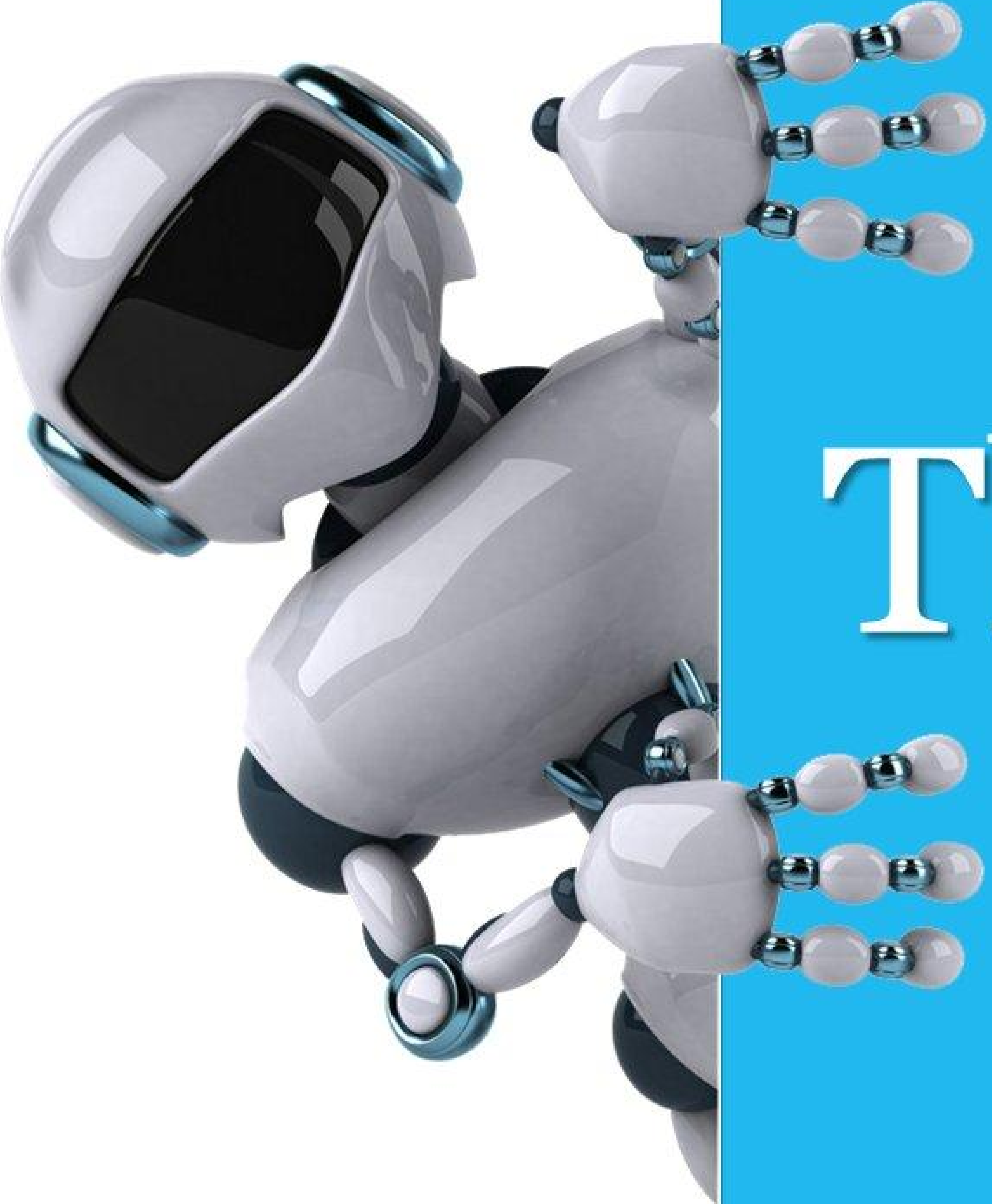
AND gate using Perceptron Model



Characteristics and Limitations of Perceptron

- ❖ **Characteristics of Perceptron:**
- ❖ Perceptron is a machine learning algorithm for **supervised learning of binary classifiers.**
- ❖ In Perceptron, the **weight coefficient is automatically learned.**
- ❖ Initially, **weights are multiplied with input features, and the decision is made whether the neuron is fired or not.**
- ❖ The **activation function applies a step rule to check whether the weight function is greater than zero.**
- ❖ The **linear decision boundary is drawn, enabling the distinction between the two linearly separable classes +1 and -1.**
- ❖ **If the added sum of all input values is more than the threshold value, it must have an output signal; otherwise, no output will be shown.**

- ❖ **Limitations of Perceptron Model**
- ❖ The **output of a perceptron can only be a binary number (0 or 1) due to the hard limit transfer function.**
- ❖ Perceptron can **only be used to classify the linearly separable sets of input vectors.**
- ❖ **If input vectors are non-linear, it is not easy to classify them properly.**



Thank you