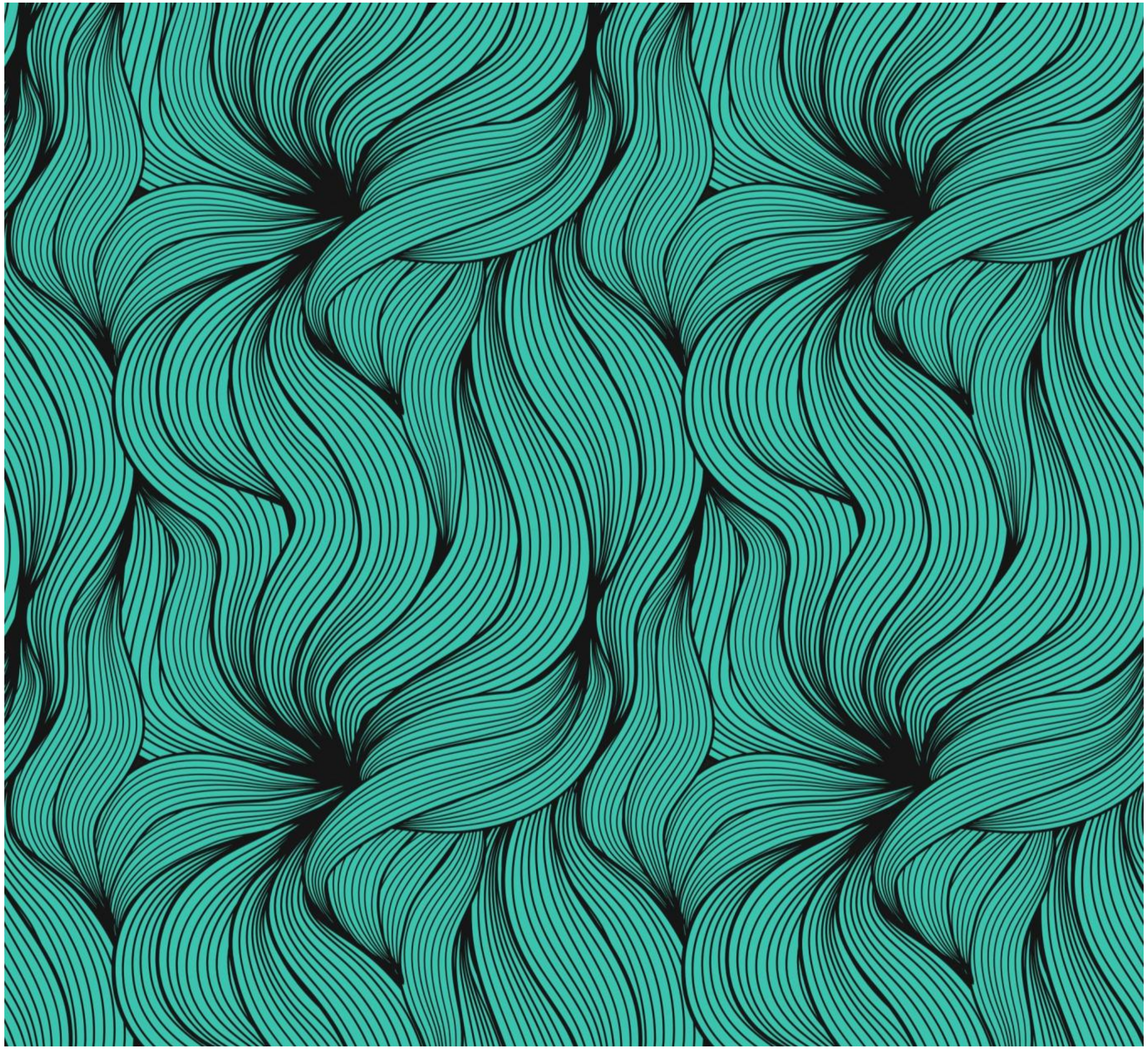
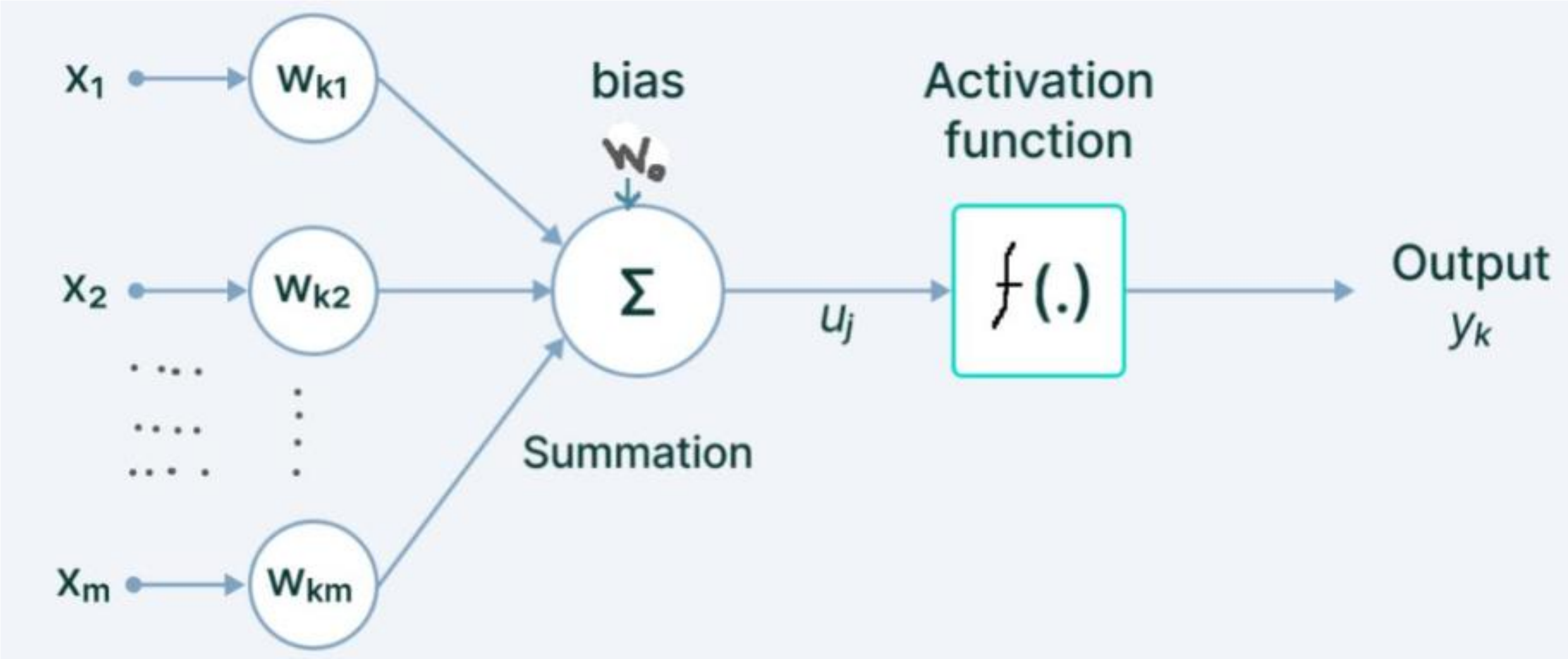
Simple ANN

Alina, Aliyanur, Tomiris



ANN – Artificial Neural Network

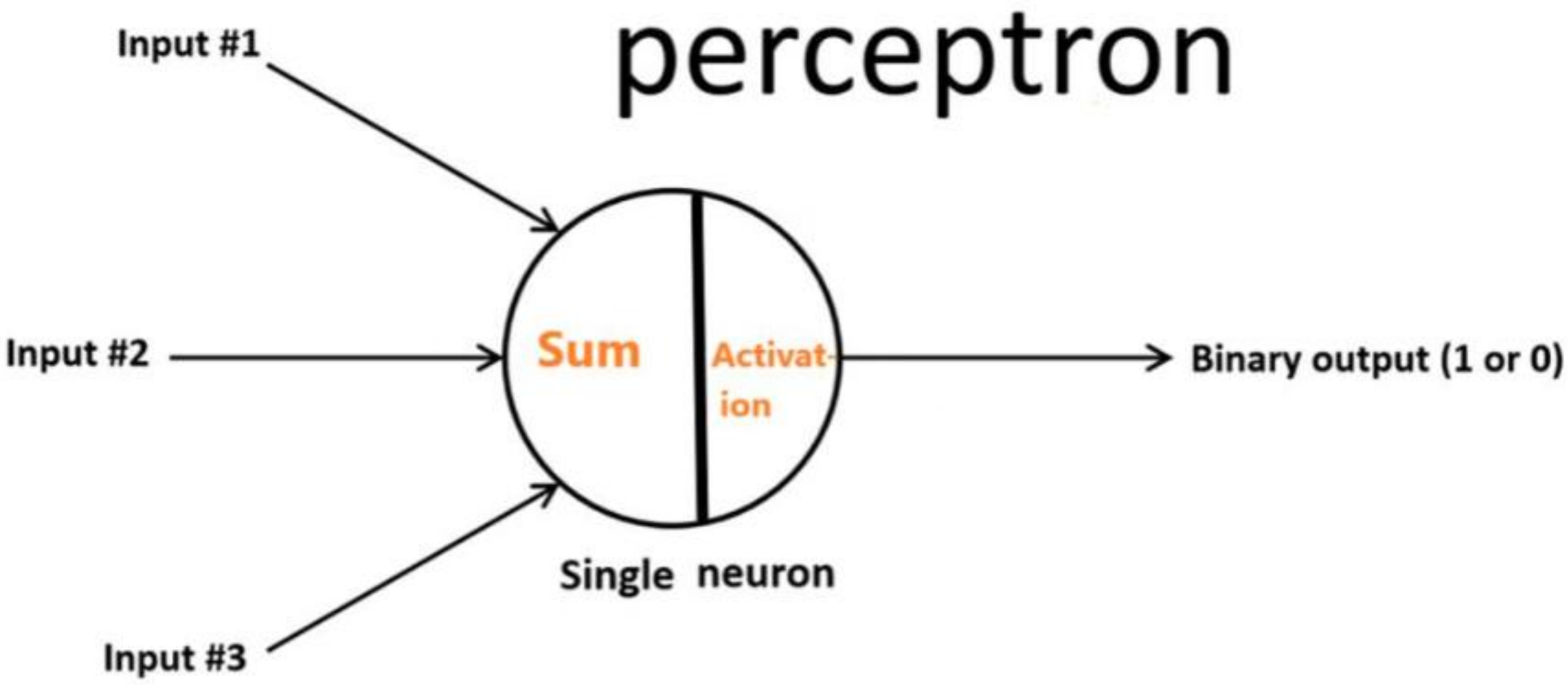
- An **artificial neuron** is a **mathematical model inspired by a biological neuron** that receives multiple input signals, multiplies them by weights, adds a bias, and applies an activation function to produce an output.



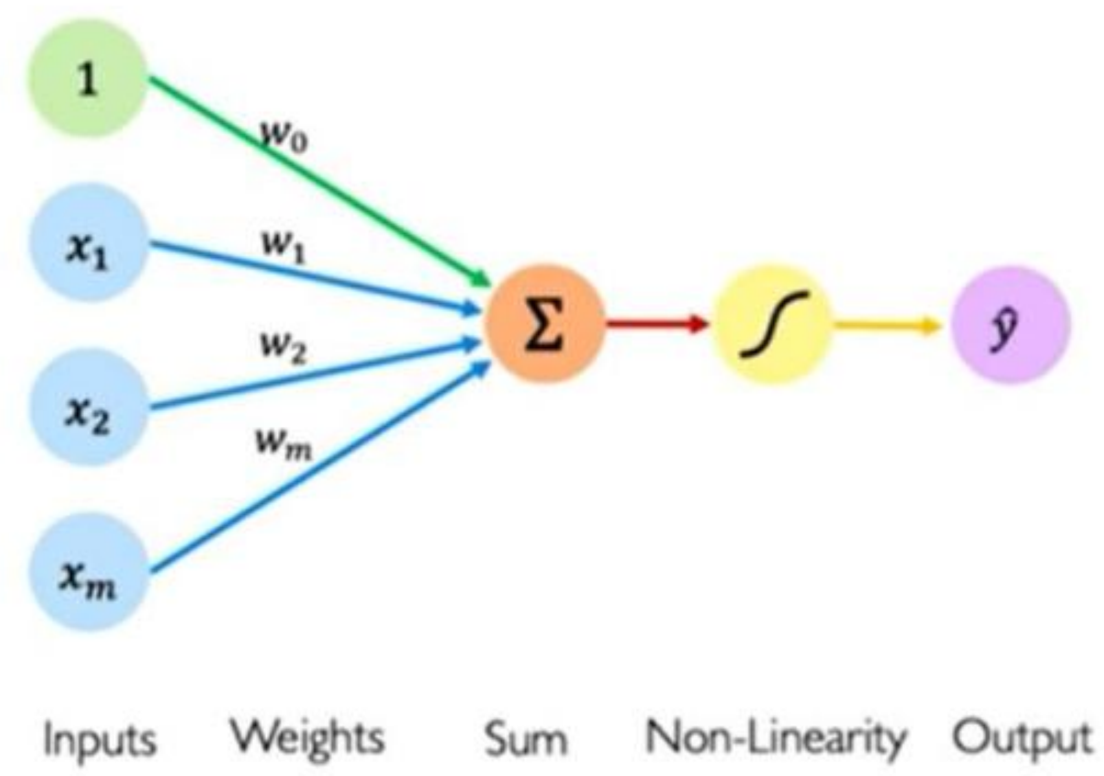
Components of an Artificial neuron

- Inputs: They are usually represented as features of a dataset which are passed on to a neural network to make predictions.
- Weights: These are the real values associated with the features. They are significant as they tell the importance of each feature which is passed as an input to the ANN.
- Bias: Bias in a neural network is required to shift the activation function across the plane either towards the left or the right. We will cover it in more detail later.
- Summation function: It is defined as the function which sums up the product of the weight and the features with bias.
- Activation function: It is required to add non-linearity to the neural network model.

In artificial neural networks, each neuron forms a weighted sum of its inputs and passes the resulting scalar value through a function referred to as an activation function to produce an output.



Four main parameters



1. The perceptron model begins with multiplying all input values (Xs) and their weights (Ws).
2. Add these values to create the weighted sum.
3. Add a bias value (W0) to this weighted sum to improve the model’s performance.
4. This weighted sum is applied to the activation function ‘f()’ to obtain the desired output.

Math model

x_i — *input* (the i-th input signal / feature)

w_i — *weight* of the i-th input (importance of x_i)

$\sum w_i x_i$ — *weighted sum of inputs*

w_0 — *bias* (also called threshold; shifts activation)

$f(\cdot)$ — *activation function* (e.g., step, sigmoid, ReLU)

$\hat{y}$ — *output* of the artificial neuron (predicted output)

i — index of inputs

$$\hat{y} = f\left(\sum w_i x_i + w_0\right)$$

Calculating weight for XOR gate

XOR		
A	B	A⊕B
0	0	0
1	0	1
0	1	1
1	1	0

Formulas:

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = \eta(t - o)x_i$$

$$y_o = \sum_{i=0}^{2} w_i x_i$$

$$if\ y_o > 1 \quad \rightarrow ouput = 1$$

$$if\ y_o \leq 1 \quad \rightarrow output = 0$$

Single perceptron doesn’t work here, need 2 layers to find a weights.

Error occurs when A = 1, B = 1.

This means we will have to combine 2 perceptrons:

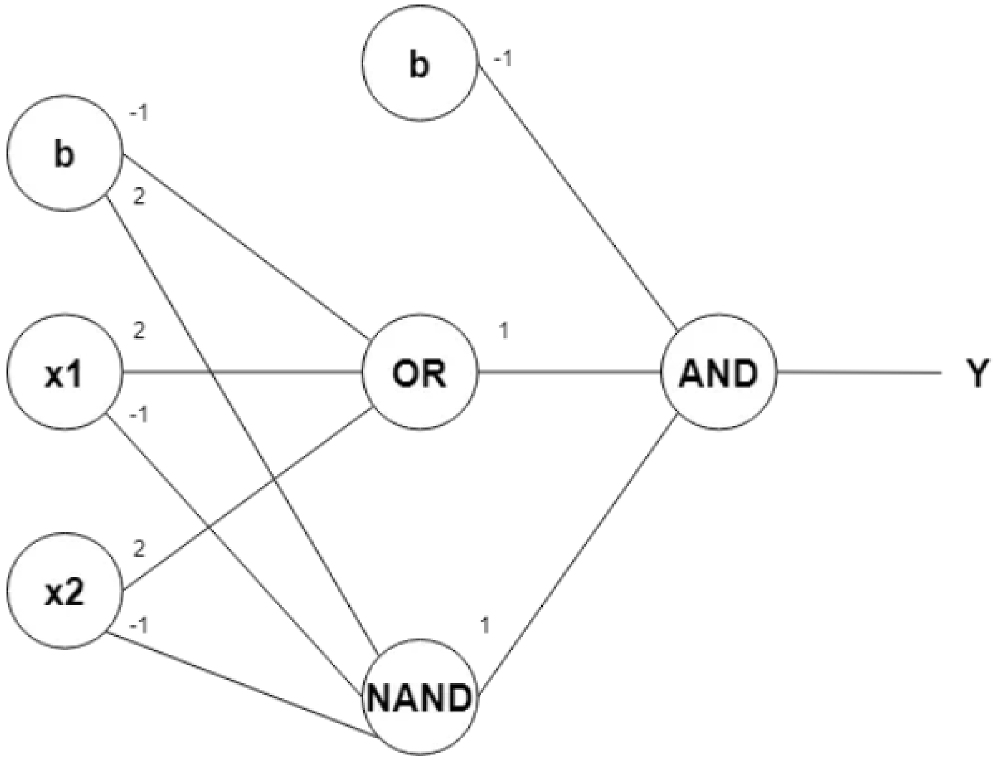
- OR (2x1+2x2−1)
- NAND (-x1-x2+2)
- AND (x1+x2−1)

The boolean representation of an XOR gate is;

> $x1x`2 + x`1x2$ Where '`' means inverse.

We first simplify the boolean expression

> $x`1x2 + x1x`2 + x`1x1 + x`2x2$
>
> $x1(x`1 + x`2) + x2(x`1 + x`2)$
>
> $(x1 + x2)(x`1 + x`2)$
>
> $(x1 + x2)(x1x2)`$



Hidden neuron 1 — OR

$$h_1 = f(x_1 + x_2 - 0.5)$$

Weights:

- $w_{11} = 1$
- $w_{12} = 1$
- bias $b_1 = -0.5$

Hidden neuron 2 — AND

$$h_2 = f(x_1 + x_2 - 1.5)$$

Weights:

- $w_{21} = 1$
- $w_{22} = 1$
- bias $b_2 = -1.5$

Output neuron — XOR

$$\boxed{y = f(h_1 - 2h_2 - 0.5)}$$

Weights:

- $w_1 = 1$ (from h_1)
- $w_2 = -2$ (from h_2)
- bias $b = -0.5$

Output neuron XOR

$$y = f(h_1 - 2h_2 - 0.5)$$

Weights:

- $w_1 = 1$ (from h_1)
- $w_2 = -2$ (from h_2)
- bias $b = -0.5$

THANKS!