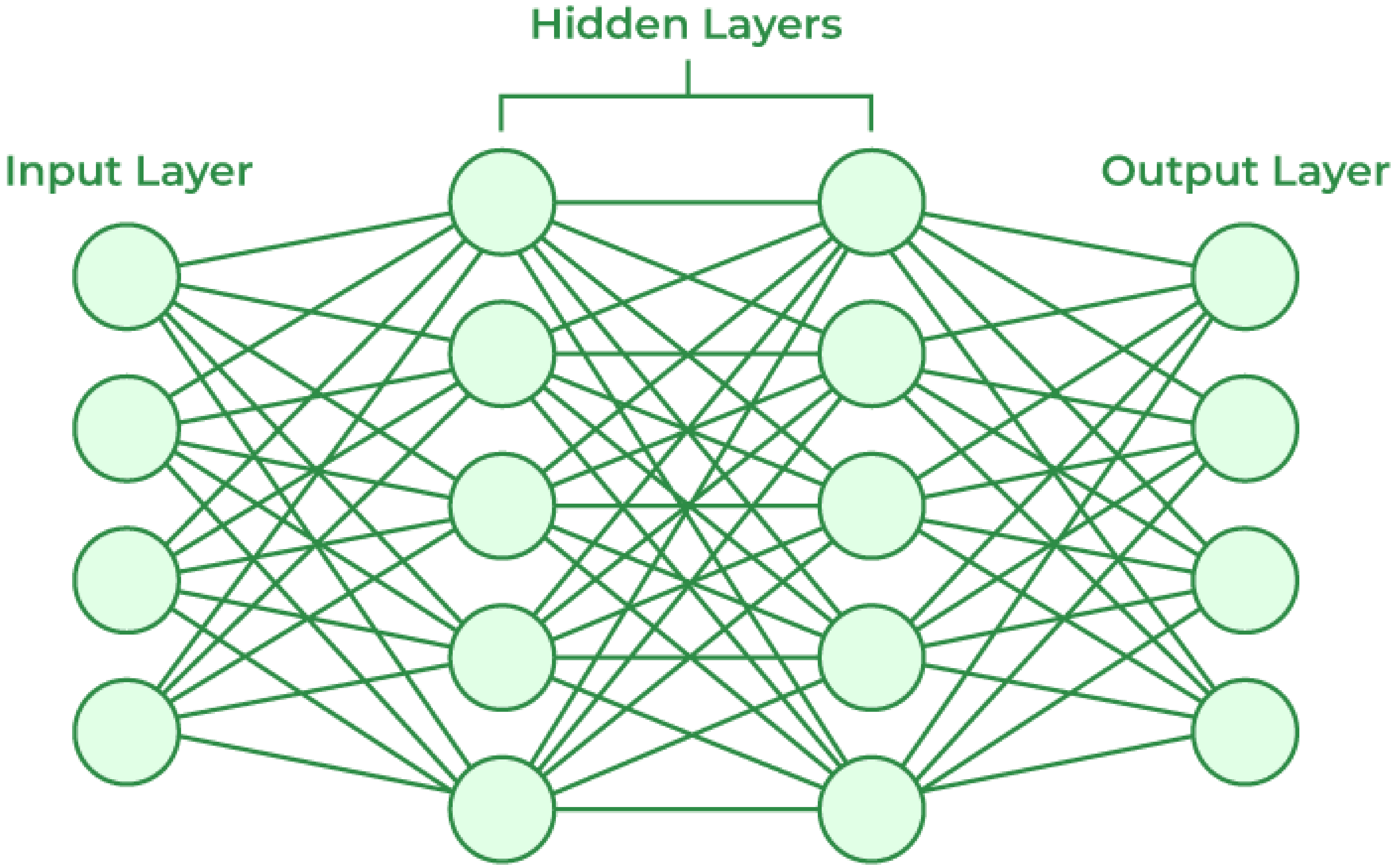
# UNIT - 4

## NEURAL NETWORKS

## UNIT IV NEURAL NETWORKS 9

Multilayer perceptron, activation functions, network training – gradient descent optimization – Stochastic gradient descent, error backpropagation, from shallow networks to deep networks – Unit saturation (aka the vanishing gradient problem) – ReLU, hyperparameter tuning, batch normalization, regularization, dropout

# NEURAL NETWORKS



# NEURAL NETWORKS

- A neural network is a method in artificial intelligence that **teaches computers to process data** in a way that is **inspired by the human brain**.
- It is a type of machine learning process, called **Deep Learning**, that uses interconnected nodes or neurons in a layered structure that resembles the human brain.
- It creates an **adaptive system** that computers use to **learn from their mistakes and improve continuously**.

# WHY ARE NEURAL NETWORKS IMPORTANT?

- Neural networks can help computers make **intelligent decisions** with **limited human assistance**.
- This is because they can learn and model the **relationships between input and output data** that are **nonlinear and complex**.

# WHY ARE NEURAL NETWORKS IMPORTANT?

**Make generalizations and inferences**

- Neural networks can comprehend **unstructured data** and make **general observations without explicit training**.
- For instance, they can recognize that two different input sentences have a similar meaning:
- **Can you tell me how to make the payment?**
- **How do I transfer money?**
- A neural network would know that both sentences mean the same thing. Or it would be able to broadly recognize that Baxter Road is a place, but Baxter Smith is a person's name.

# EVOLUTION OF NEURAL NETWORKS

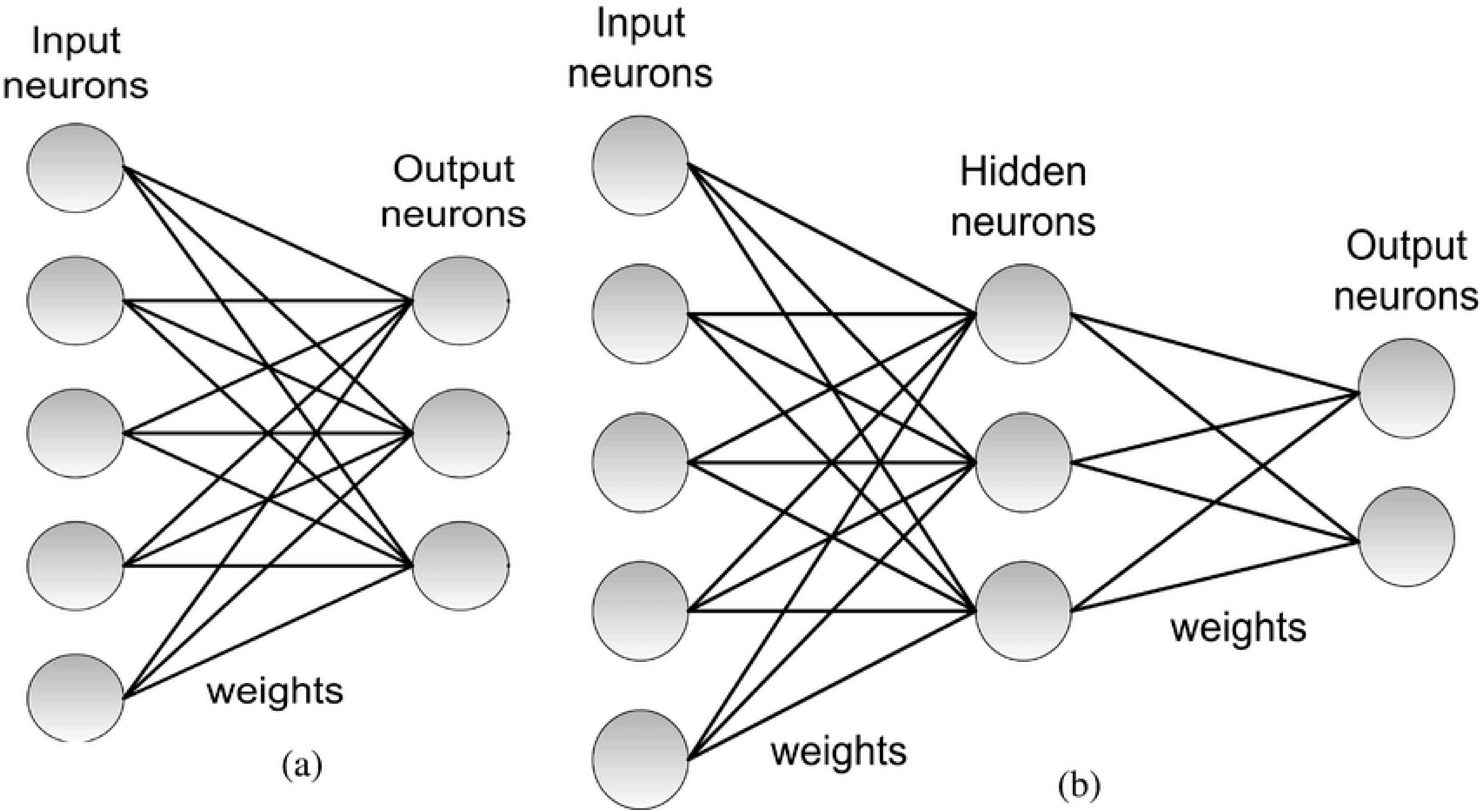
## 1940s-1950s: Early Concepts

- Neural networks began with the introduction of the first mathematical model of **artificial neurons** by McCulloch and Pitts.
- But **computational constraints** made progress difficult.

## 1960s-1970s: Perceptrons

- This era is defined by the work of Rosenblatt on perceptrons. <u>Perceptrons</u> are **single-layer networks** whose applicability was limited to issues that could be solved **linearly separately**.

# SINGLE LAYER PERCEPTRON



# EVOLUTION OF NEURAL NETWORKS

## 1980s: Backpropagation and Connectionism

- **Multi-layer network** training was made possible by Rumelhart, Hinton, and Williams' invention of the backpropagation method. With its emphasis on learning through interconnected nodes, connectionism gained appeal.

## 1990s: Boom and Winter

- With applications in image identification, finance, & other fields, neural networks saw a boom. Neural network research did, however, experience a "**winter**" due to exorbitant computational costs and inflated expectations.

# WHY IS THE PERIOD FROM 1974-93 KNOWN AS AI WINTER?

- In 1974, in response to the criticism from James Lighthill and ongoing pressure from congress, the U.S. and British Governments **stopped funding undirected research into artificial intelligence**, and the difficult years that followed would later be known as an "**AI winter**".

# EVOLUTION OF NEURAL NETWORKS

## 2000s: Resurgence and Deep Learning

- Deep learning has shown amazing effectiveness in a number of disciplines by utilizing numerous layers.

## 2010s-Present: Deep Learning Dominance

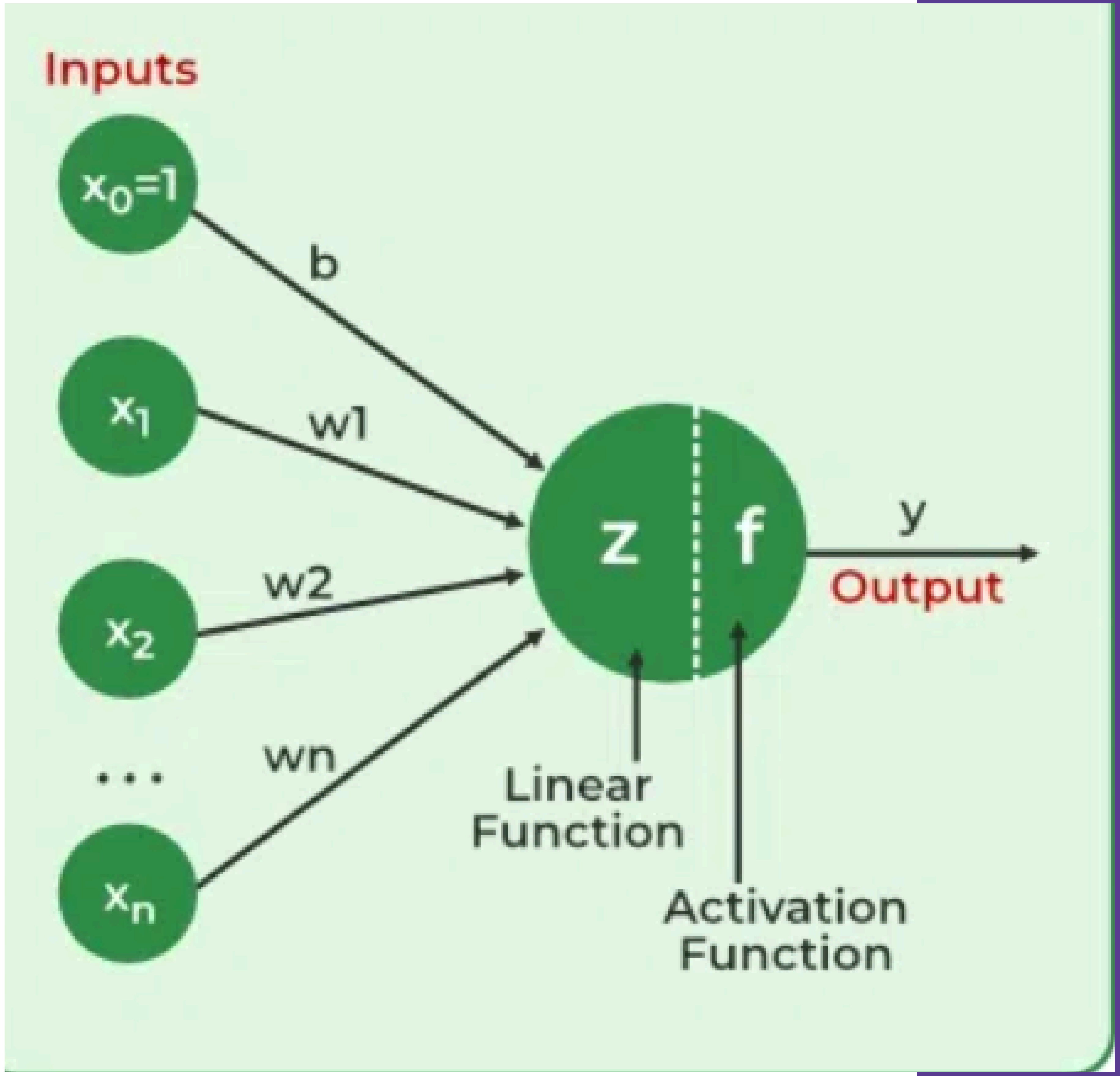
- Convolutional neural networks (CNNs) and recurrent neural networks (RNNs), two deep learning architectures, dominated machine learning. Their power was demonstrated by innovations in gaming, picture recognition, and natural language processing.

# COMPONENTS OF NEURAL NETWORKS

- Network components include neurons, connections, weights, biases, propagation functions, and a learning rule.
- Neurons receive inputs, governed by thresholds and activation functions.
- Connections involve weights and biases regulating information transfer.

# COMPONENTS OF NEURAL NETWORKS

- Learning, adjusting weights and biases, occurs in three stages: input computation, output generation, and iterative refinement enhancing the network’s proficiency in diverse tasks.



# HOW DOES NEURAL NETWORKS WORK?

- Consider a neural network for **email classification**.
- The input layer takes features like **email content, sender information, and subject**.
- These inputs, multiplied by adjusted weights, pass through hidden layers.
- The network, through training, learns to recognize patterns indicating whether an email is spam or not.

# HOW DOES NEURAL NETWORKS WORK?

- The output layer, with a binary activation function, predicts whether the email is spam (1) or not (0).
- As the network iteratively refines its weights through backpropagation, it becomes adept at distinguishing between spam and legitimate emails, showcasing the practicality of neural networks in real-world applications like email filtering.

# WORKING OF A NEURAL NETWORK

- Neural networks are complex systems that mimic some features of the functioning of the human brain.
- It is composed of an input layer, one or more hidden layers, and an output layer made up of layers of artificial neurons that are coupled.
- The two stages of the basic process are called **backpropagation and forward propagation**.

# FORWARD PROPAGATION

- **Input Layer:** Each feature in the input layer is represented by a node on the network, which receives input data.
- **Weights and Connections:** The weight of each neuronal connection indicates how strong the connection is.
- Throughout training, these weights are changed.

# FORWARD PROPAGATION

- **Hidden Layers:** Each hidden layer neuron processes inputs by multiplying them by weights, adding them up, and then passing them through an activation function.
- By doing this, **non-linearity** is introduced, enabling the network to recognize intricate patterns.
- **Output:** The final result is produced by repeating the process until the output layer is reached.

# BACKPROPAGATION

1. **Loss Calculation:** The network's output is evaluated against the real goal values, and a loss function is used to compute the difference.
    - For a regression problem, the Mean Squared Error (MSE) is commonly used as the cost function.
2. **Loss Function:**
    - Gradient Descent: Gradient descent is then used by the network to reduce the loss. To lower the inaccuracy, weights are changed based on the derivative of the loss with respect to each weight.

# BACKPROPAGATION

**3. Adjusting weights:** The weights are adjusted at each connection by applying this iterative process, or backpropagation, backward across the network.

**4. Training:** During training with different data samples, the entire process of forward propagation, loss calculation, and backpropagation is done iteratively, enabling the network to adapt and learn patterns from the data.

**5. Activation Functions:** Model non-linearity is introduced by activation functions like the Rectified Linear Unit (ReLU) or sigmoid. Their decision on whether to "fire" a neuron is based on the whole weighted input.

# Multilayer Perceptron

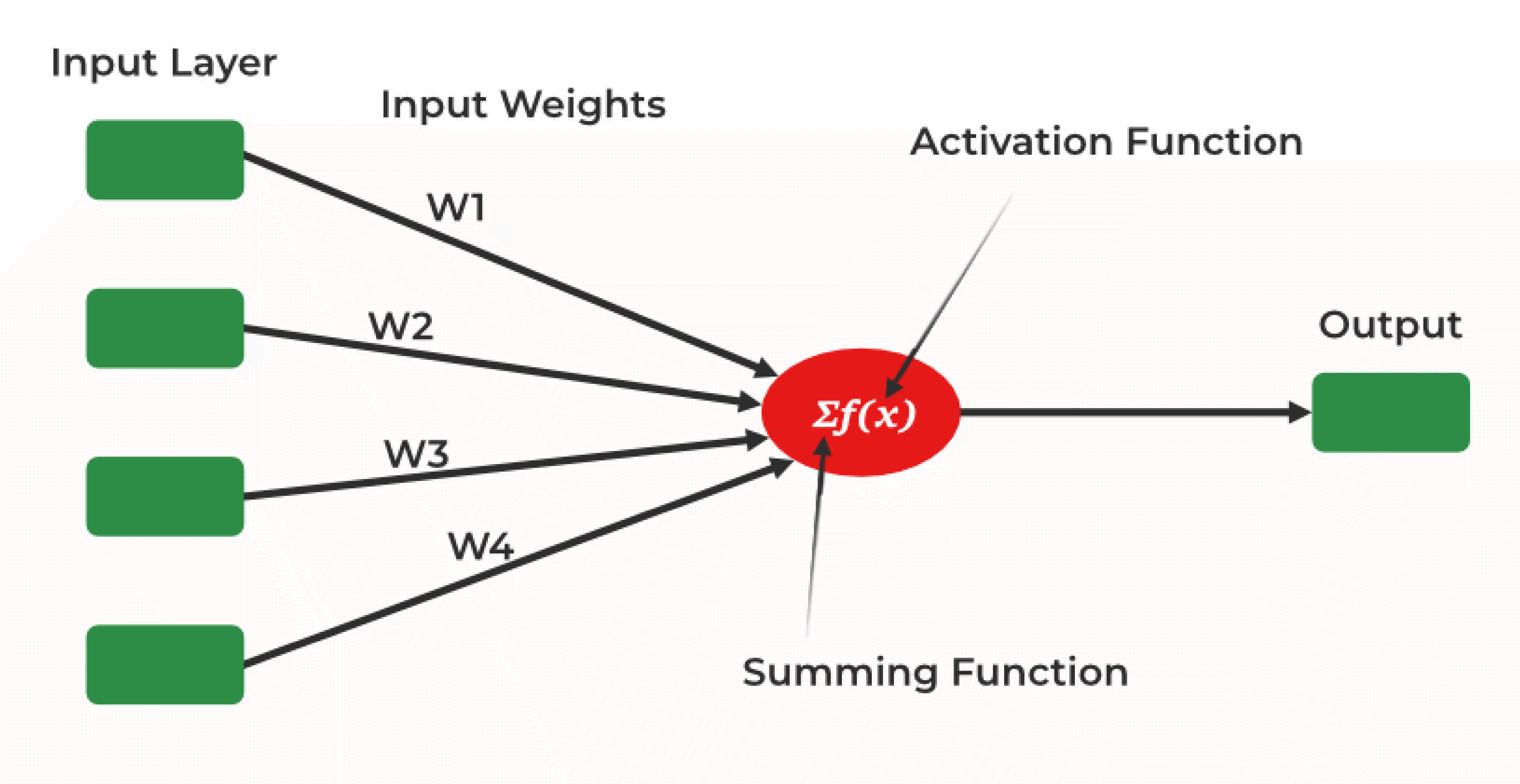
# Single Layer Perceptron

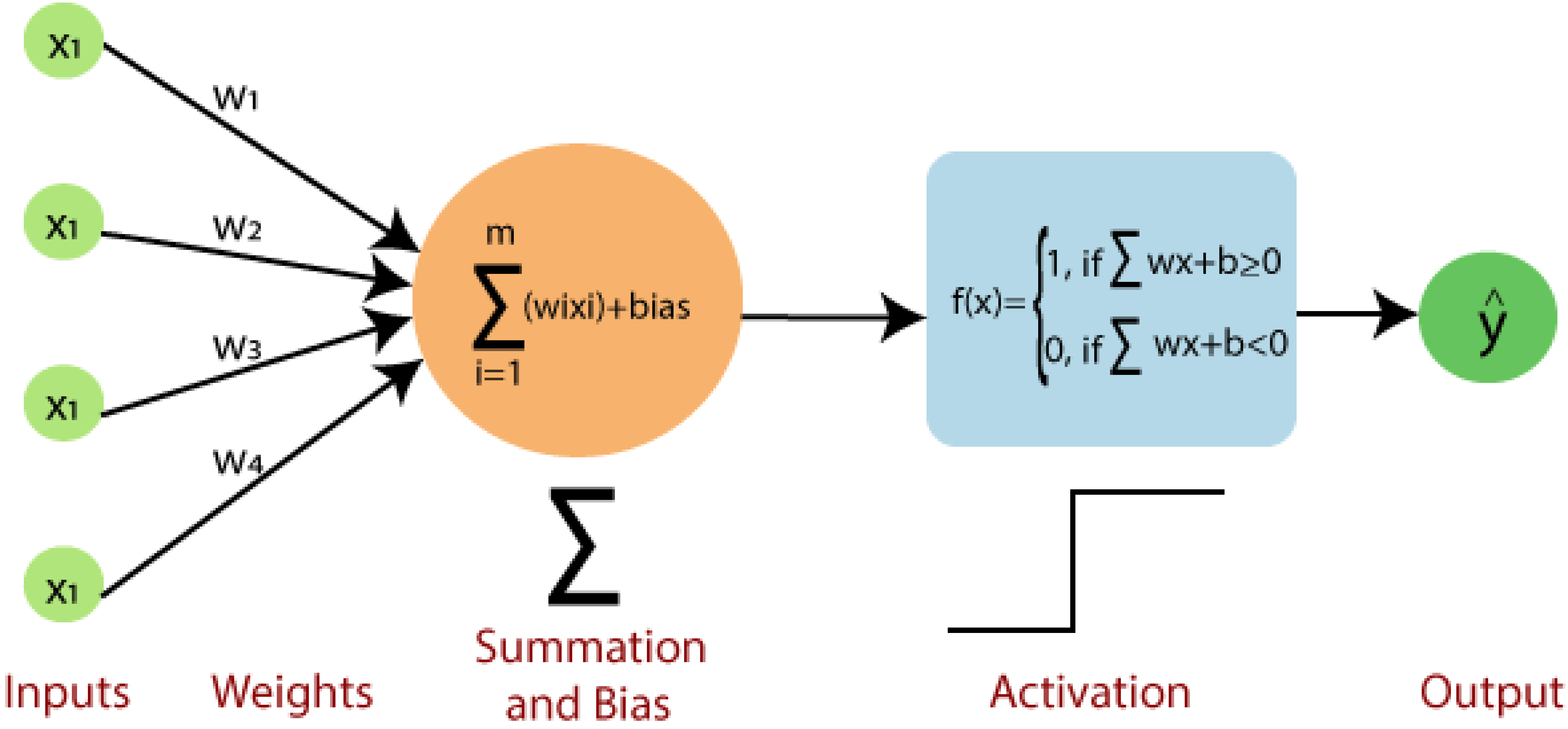
- It is one of the oldest and first introduced neural networks.
- It was proposed by Frank Rosenblatt in 1958.
- Perceptron is also known as an Artificial Neural Network (ANN).
- Perceptron is mainly used to compute the logical gate like AND, OR, and NOR which has binary input and binary output.
- Perceptron is mainly used to classify the data into two parts. Therefore, it is also known as Linear Binary Classifier.

The main functionality of the perceptron is:-

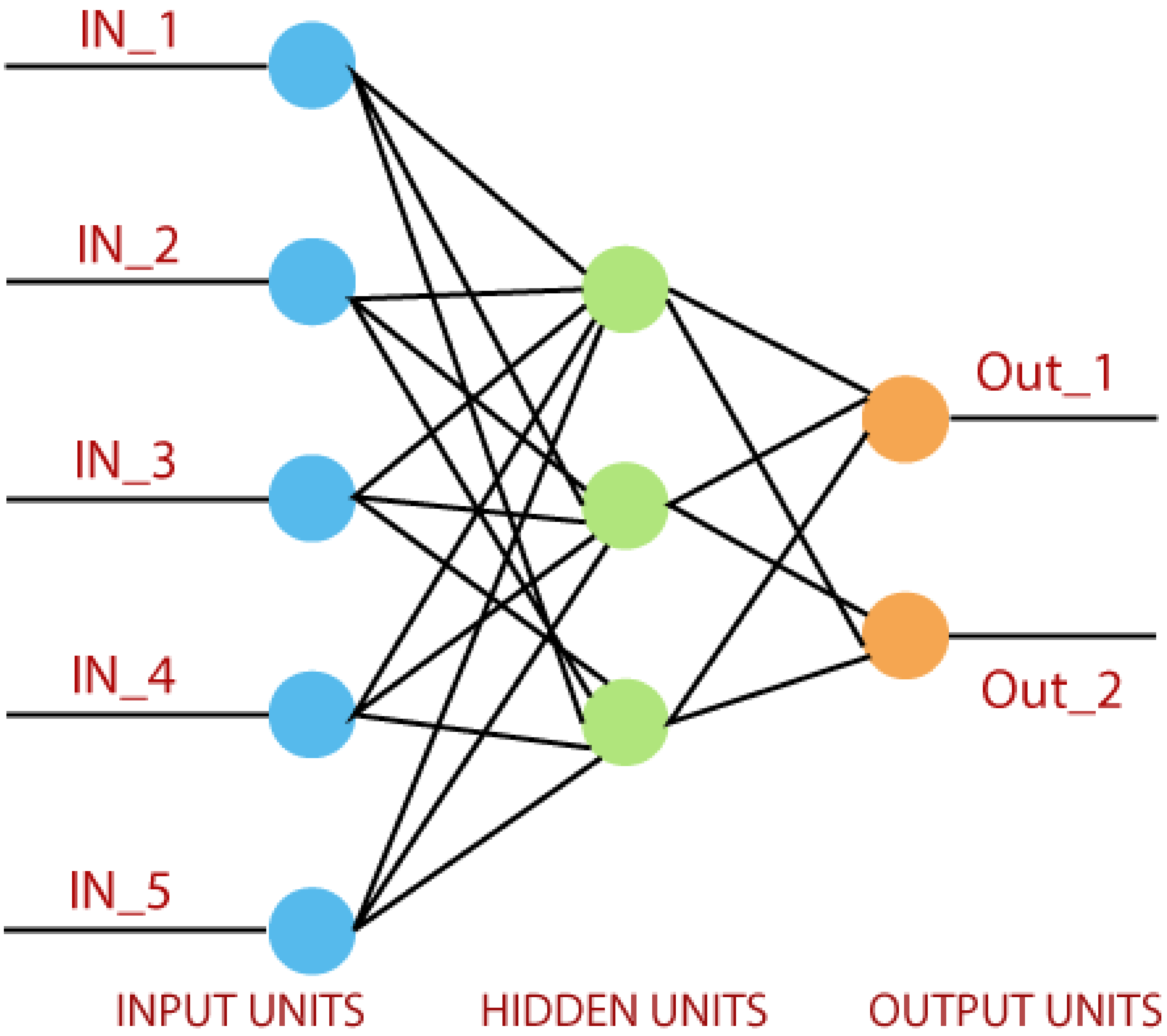
- Takes input from the input layer
- Weight them up and sum it up
- Pass the sum to the nonlinear function to produce the output

# SINGLE-LAYER NEURAL NETWORK





x1
x1
x1
x1
w1
w2
w3
w4
m
∑ (wixi)+bias
i=1
f(x)= 1, if ∑ wx+b≥0
0, if ∑ wx+b<0
ŷ
∑
Summation and Bias
Inputs
Weights
Activation
Output


IN_1
IN_2
IN_3
IN_4
IN_5
Out_1
Out_2
INPUT UNITS
HIDDEN UNITS
OUTPUT UNITS

# Multi Layer Perceptron

- Multi-layer perception is also known as MLP.
- It is fully connected dense layers, which transform any input dimension to the desired dimension.
- A multi-layer perception is a neural network that has multiple layers.
- A multi-layer perceptron has one input layer and for each input, there is one neuron(or node), it has one output layer with a single node for each output and it can have any number of hidden layers and each hidden layer can have any number of nodes.

- MLP belongs to the feedforward neural network.
- MLP works only in the forward direction.
- It is an Artificial Neural Network in which all nodes are interconnected with nodes of different layers.
- Each node passes its value to the coming node only in the forward direction.
- The MLP neural network uses a Backpropagation algorithm to increase the accuracy of the training model.

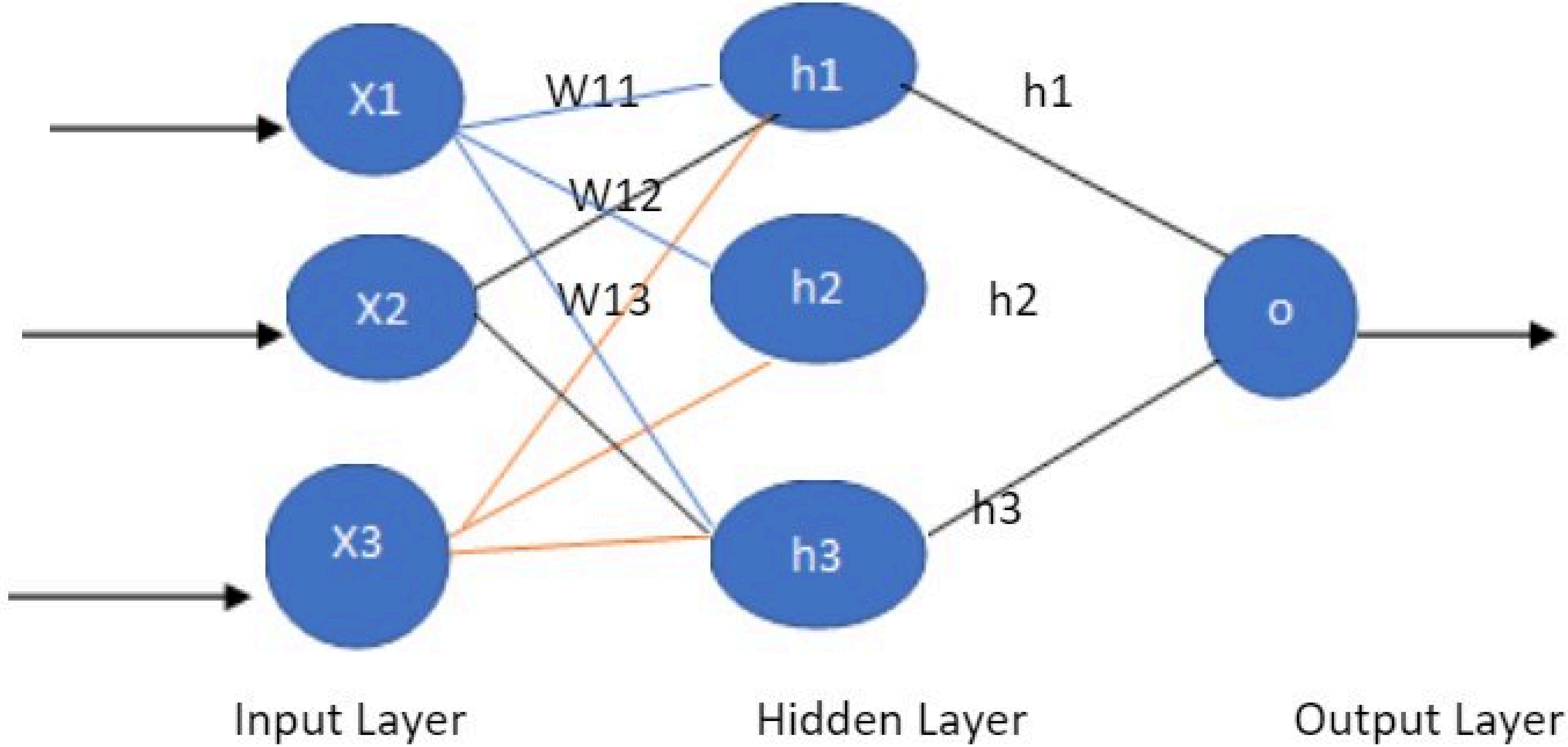



Diagram Of MultiLayer Perceptron Neural Network

## Working of MultiLayer Perceptron Neural Network

- The input node represents the feature of the dataset.
- Each input node passes the vector input value to the hidden layer.
- In the hidden layer, each edge has some weight multiplied by the input variable.
- All the production values from the hidden nodes are summed together.

- The activation function is used in the hidden layer to identify the active nodes.
- The output is passed to the output layer.
- Calculate the difference between predicted and actual output at the output layer.
- The model uses backpropagation after calculating the predicted output.

## Advantages of MultiLayer Perceptron Neural Network:

- MLP Neural Networks can easily work with non-linear problems.
- It can handle complex problems while dealing with large datasets.
- It has a higher accuracy rate and reduces prediction error by using backpropagation.
- After training the model, the MLP Neural Network quickly predicts the output.

## Disadvantages of MultiLayer Perceptron Neural Network:

- This Neural Network consists of large computation, which sometimes increases the overall cost of the model.
- The model will perform well only when it is trained perfectly.
- Due to this model's tight connections, the number of parameters and node redundancy increases.

# Activation Function

# Activation Function

- Activation function is an important part of an artificial neural network.
- They basically decide whether a neuron should be activated or not.
- It also performs a nonlinear transformation on the input to get better results on a complex **neural network.**

# Why do we need Activation Functions?

- Without activation function, weight and bias would only have a linear transformation, or neural network is just a linear regression model, that cannot be able to solve complex problems.
- The addition of activation function to neural network executes the non-linear transformation to input and make it capable to solve complex problems

# Types of Activation Functions

A. Binary step function

B. Linear function

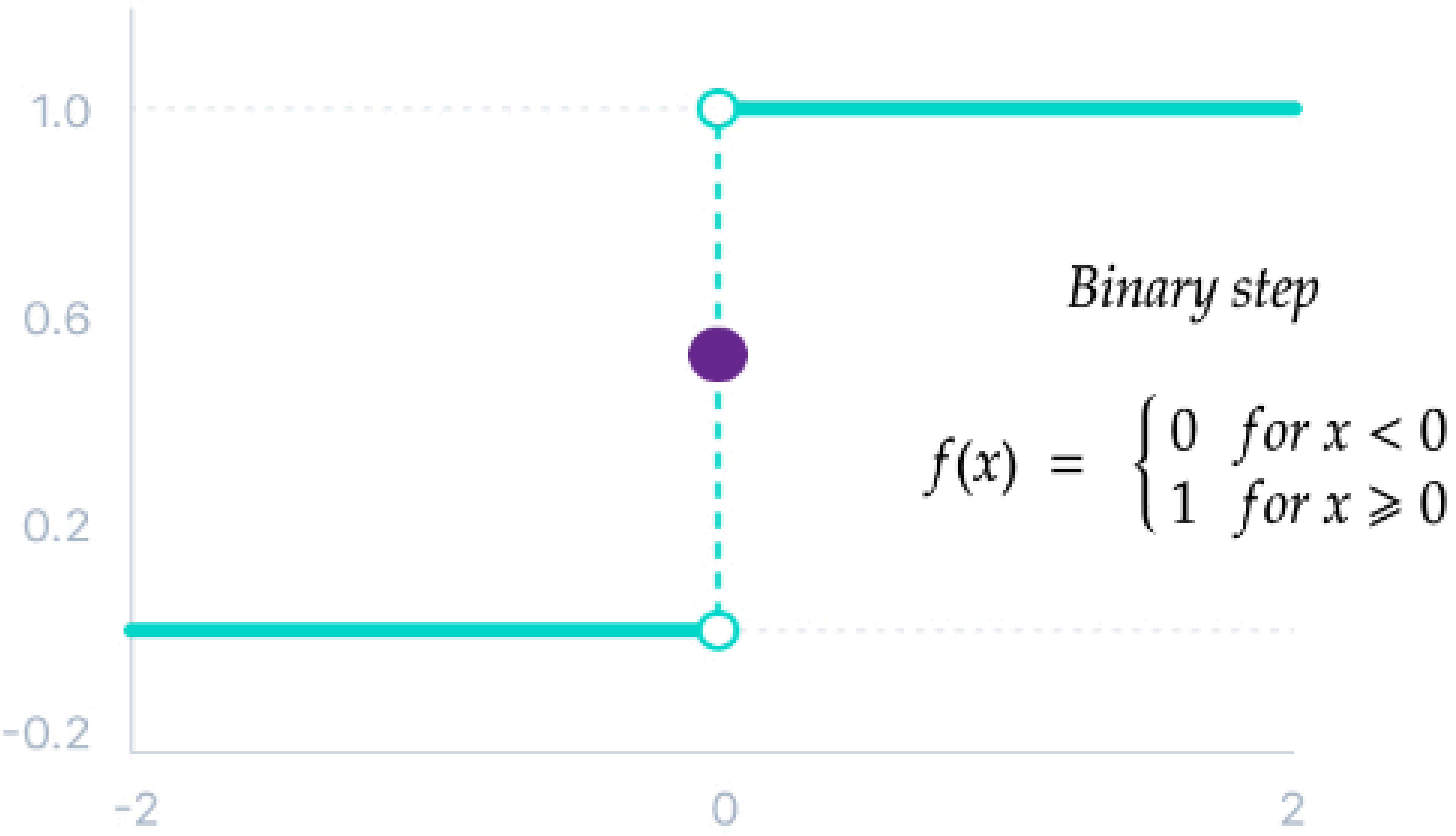
C. Non linear activation function

# Binary Step Neural Network Activation Function

- This activation function is basically a threshold base classifier.
- we decide some threshold value to decide output that neuron should be activated or deactivated.

- The input fed to the activation function is compared to a threshold; if the input is greater than it, then the neuron is activated, else it is deactivated, meaning that its output is not passed on to the next hidden layer.
- It cannot provide multi-value outputs—for example, it cannot be used for multi-class classification problems.

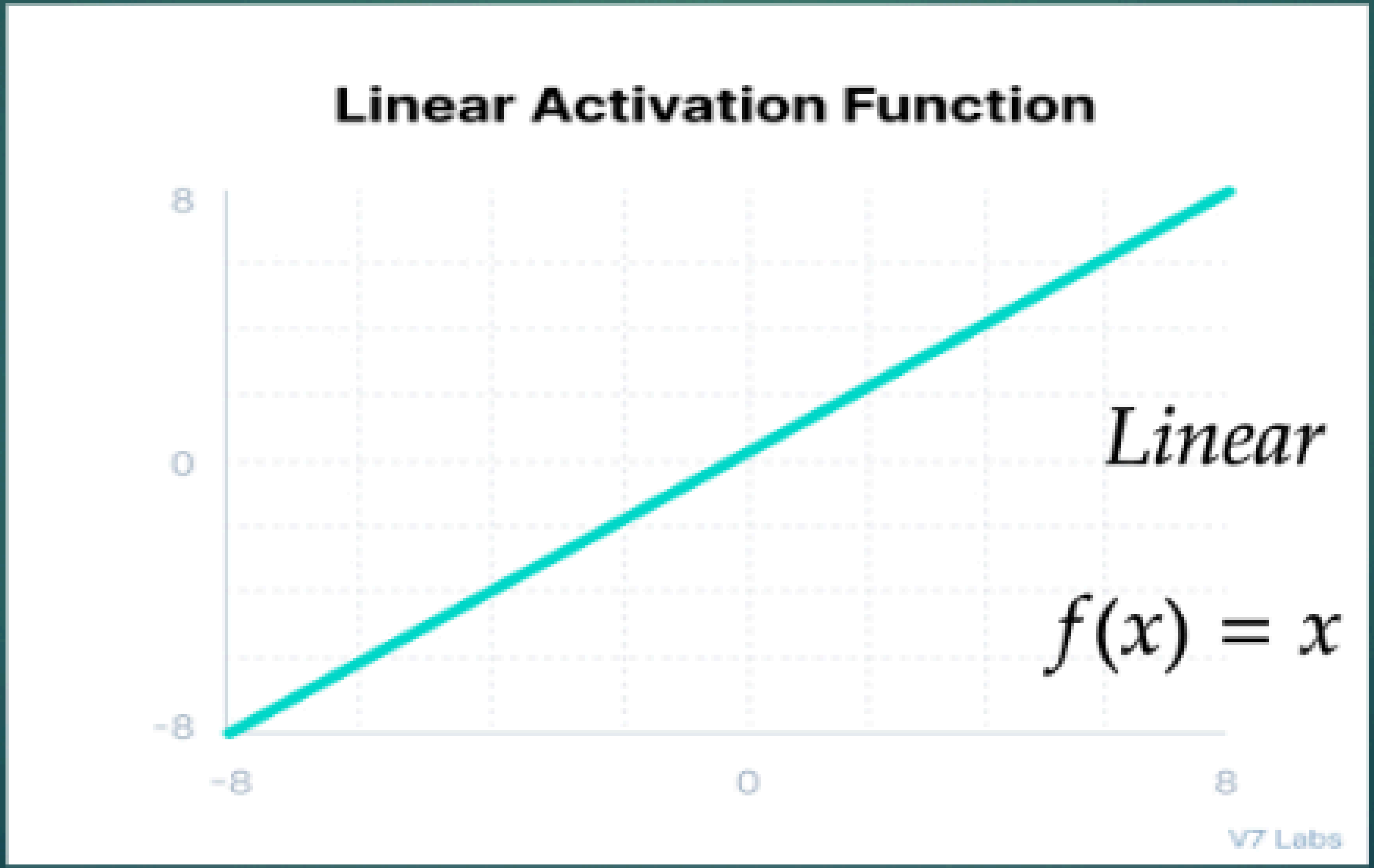
# Binary Step Function




$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

# Linear Activation Function

- The linear activation function, also known as "no activation," or "identity function" (multiplied x1.0), is where the activation is proportional to the input.



# Non Linear Activation Functions

- The linear activation function is simply a linear regression model.
- Because of its limited power, this does not allow the model to create complex mappings between the network's inputs and outputs.
- Non-linear activation functions solve the limitations of linear activation functions:
  - They allow back propagation

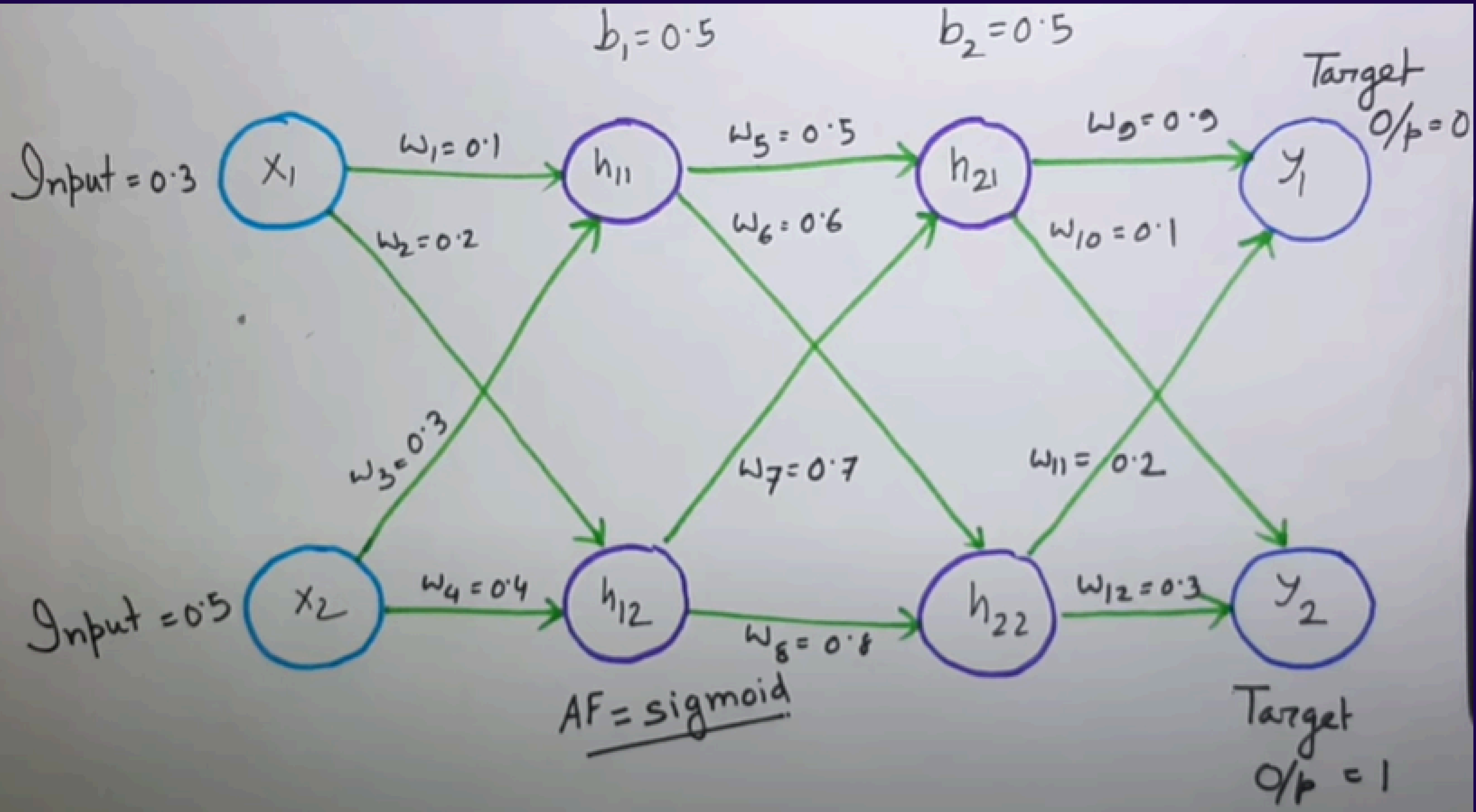
# Non-Linear Neural Networks Activation Functions

- ✓ Sigmoid / Logistic Activation Function
- ✓ Tanh Function (Hyperbolic Tangent)
- ✓ ReLU Function
- ✓ Leaky ReLU Function
- ✓ Softmax Function

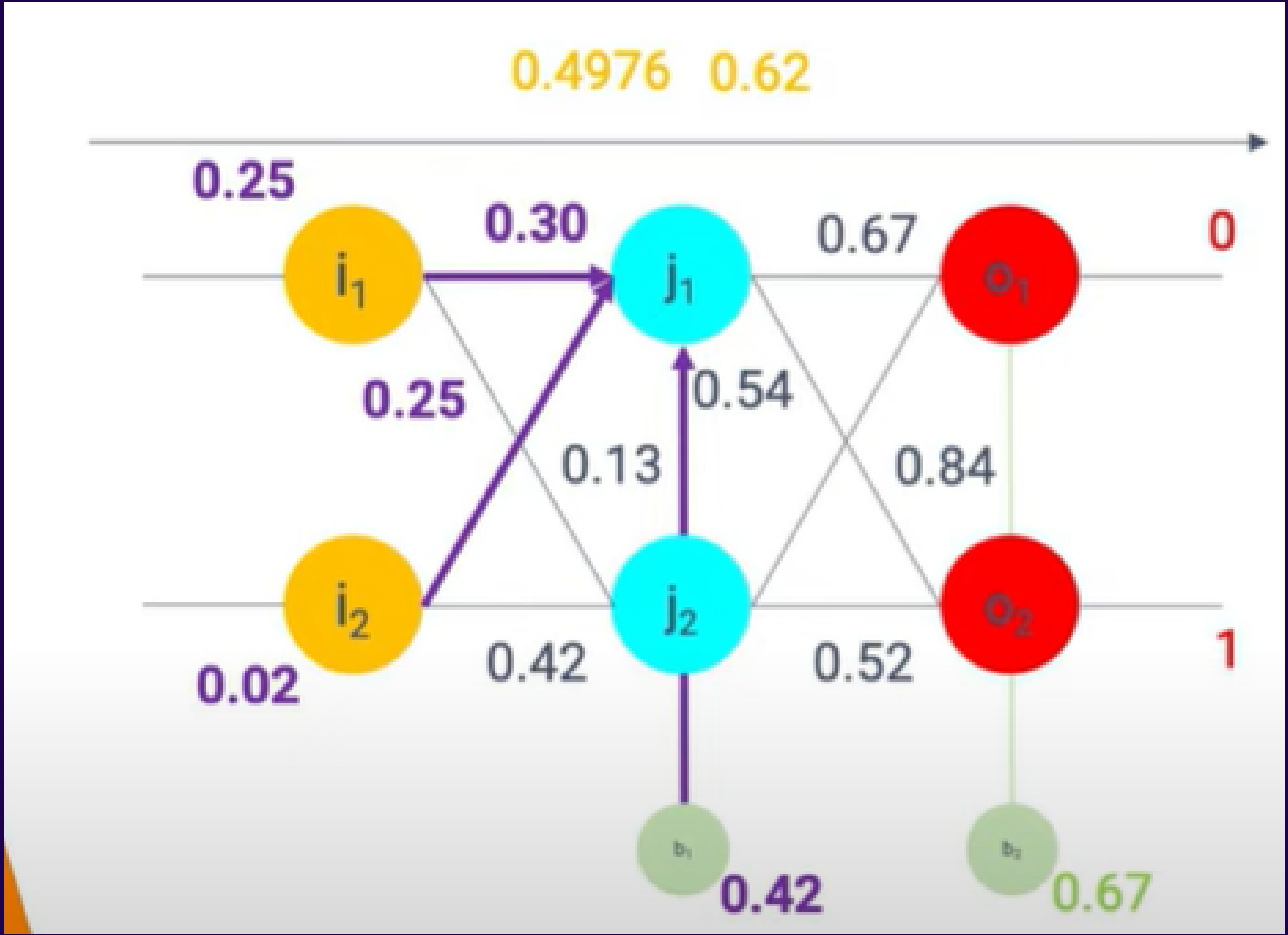
# Sigmoid / Logistic Activation Function

- This function takes any real value as input and outputs values in the range of 0 to 1.
- The larger the input (more positive), the closer the output value will be to 1.0, whereas the smaller the input (more negative), the closer the output will be to 0.0

# Feed Forward Neural Network Calculation



b1 = 0.5
b2 = 0.5
Target O/p = 0
Input = 0.3
x1
w1 = 0.1
h11
w5 = 0.5
h21
w9 = 0.9
y1
w2 = 0.2
w6 = 0.6
w10 = 0.1
w3 = 0.3
w7 = 0.7
w11 = 0.2
Input = 0.5
x2
w4 = 0.4
h12
w8 = 0.8
h22
w12 = 0.3
y2
AF = sigmoid
Target O/p = 1


0.4976 0.62
0.25
0.30
0.67
0
$i_1$
$j_1$
$o_1$
0.25
0.54
0.13
0.84
$i_2$
$j_2$
$o_2$
0.42
0.52
1
0.02
$b_1$
$b_2$
0.42
0.67

# Backward propagation

*We are looking to measure the impact of each weight on the total error.*