Course Title

ARTIFICIAL INTELLIGENT AND MACHINE LEARNING

COURSE CODE:23AD2001O

Topic

Artificial Neural Network: Introduction to ANN, McCulloch Pits, Perceptron

AIM

To familiarize students with the concept of ANN

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

1. Explain about History of ANN.
2. Explain Mc-Culloth Pitts mode;
3. Understand about Perceptron model.

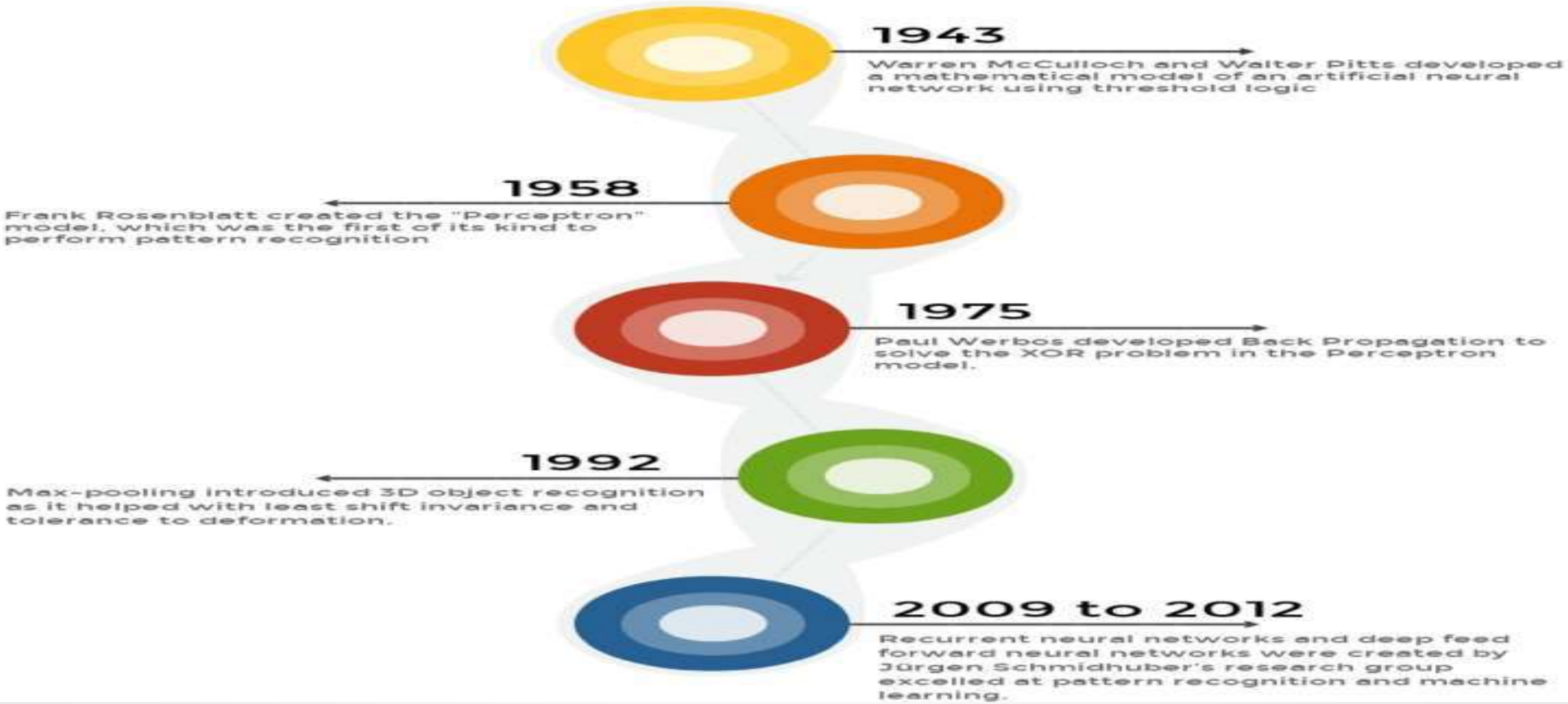
LEARNING OUTCOMES

At the end of this session, you should be able to:

1. Know history of ANN.
2. Write perceptron learning algorithms.
3. Design Neural Networks for different data.

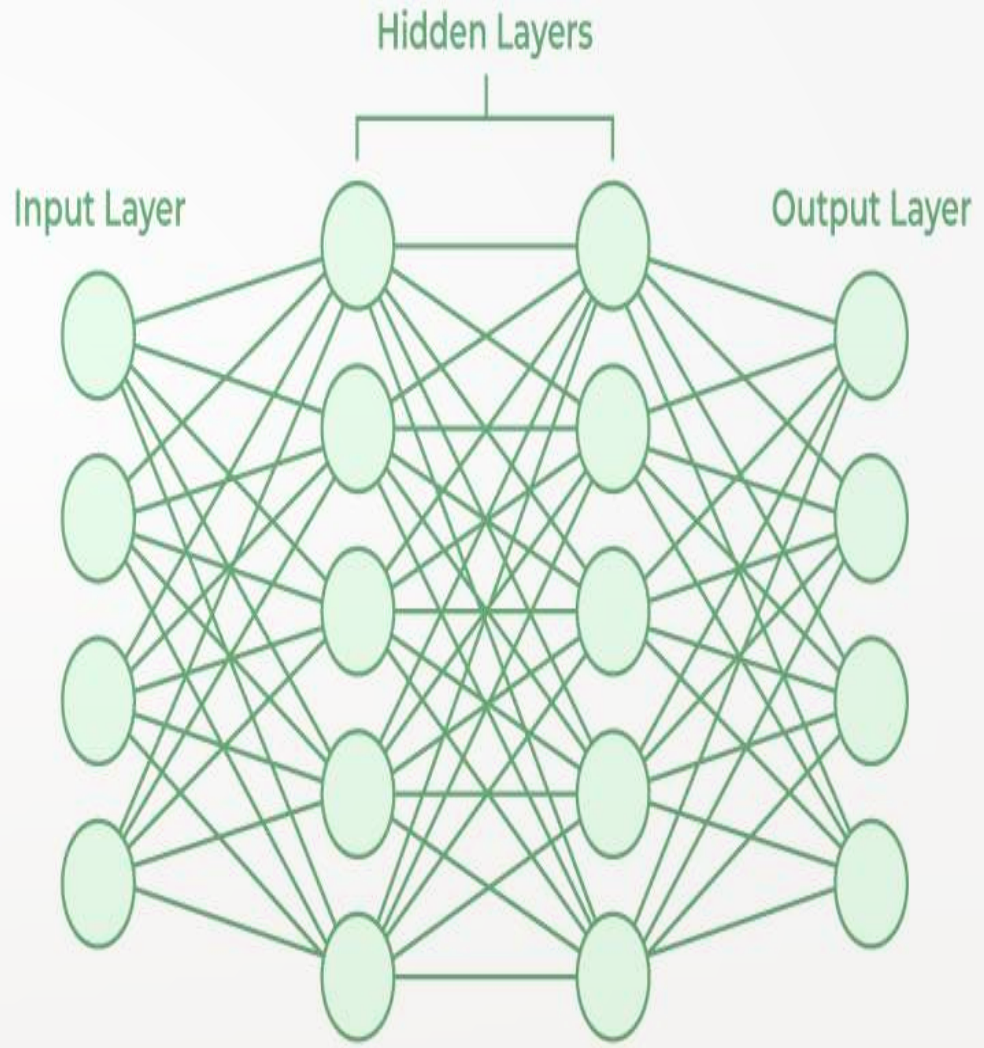
CONTENTS

The History of Neural Networks



INTRODUCTION TO ARTIFICIAL NEURAL NETWORKS

- The structures and operations of human neurons serve as the basis for artificial neural networks.
- Artificial Neural Networks contain artificial neurons which are called units.
- These units are arranged in a series of layers that together constitute the whole Artificial Neural Network in a system. Commonly, Artificial Neural Network has an input layer, an output layer as well as hidden layers.
- The input layer of an artificial neural network is the first layer, and it receives input from external sources and releases it to the hidden layer, which is the second layer.
- In the hidden layer, each neuron receives input from the previous layer neurons, computes the weighted sum, and sends it to the neurons in the next layer.
- These connections are weighted means effects of the inputs from the previous layer are optimized more or less by assigning different-different weights to each input and it is adjusted during the training process by optimizing these weights for improved model performance.



APLLICATIONS OF ANN

- Artificial Neural Networks (ANNs) are a type of machine learning model loosely inspired by the structure and function of the human brain.
- ANNs are made up of interconnected nodes, called artificial neurons, that process information. These nodes are arranged in layers, with each layer feeding information to the next.
- By adjusting the connections between these nodes, ANNs can learn to identify patterns in data and make predictions.

APPLICATIONS OF ANN

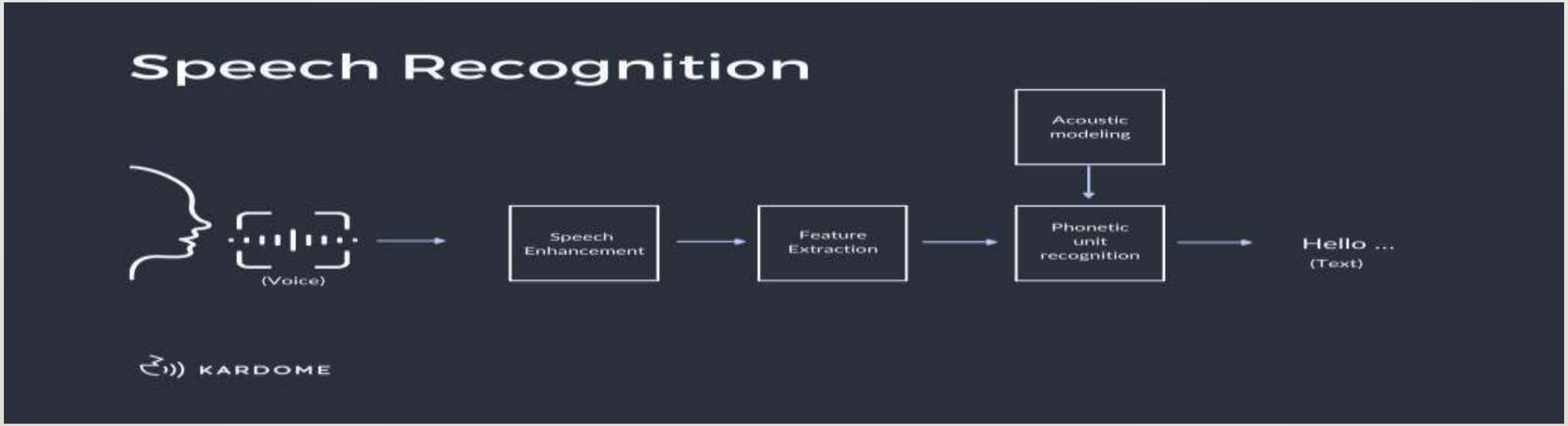
- ANNs have a wide range of applications, including:
- **Image recognition**
- Speech recognition
- Machine translation
- Medical diagnosis
- Financial forecasting
- Recommendation systems

APPLICATIONS OF ANN…

- **Image recognition:** ANNs are used in a variety of image recognition tasks, such as facial recognition, object detection, and scene classification. For example, ANNs are used to power the facial recognition features in many smartphones and social media platforms.

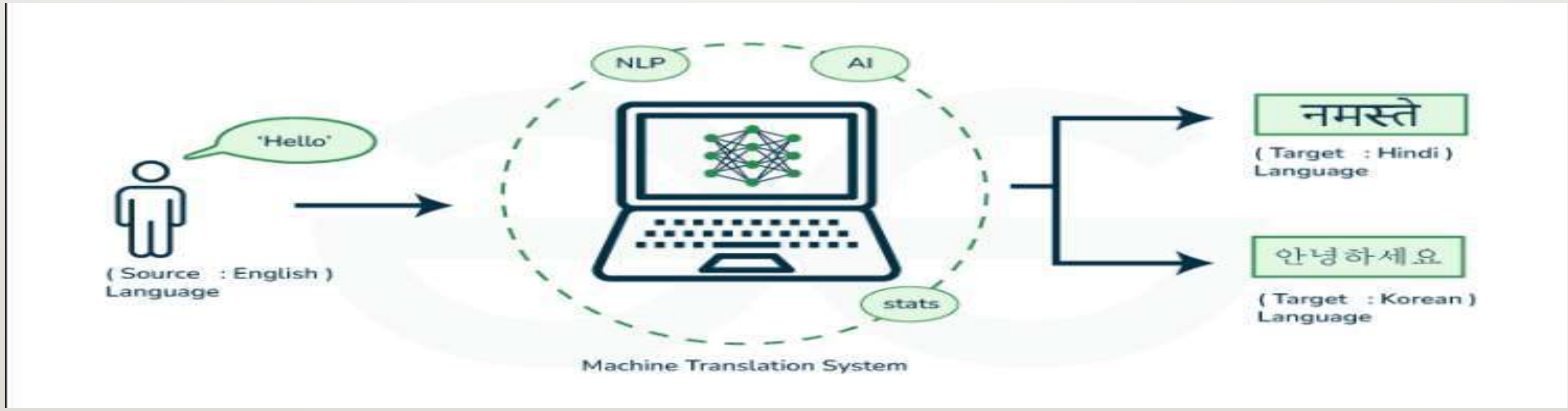
APPLICATIONS OF ANN…..

- **Speech recognition:** ANNs are also used in speech recognition tasks, such as voice assistants and automatic transcription software. For instance, ANNs are used in Amazon Alexa and Apple Siri to recognize and respond to spoken commands



APPLICATIONS OF ANN…..

- **Machine translation:** ANNs are becoming increasingly adept at machine translation, which is the task of translating text from one language to another. Machine translation powered by ANNs is used in a variety of applications, such as Google Translate and Microsoft Translator.



APPLICATIONS OF ANN…..

- **Medical diagnosis:** ANNs are being used to develop medical diagnostic tools that can analyze medical images and patient data to identify diseases and disorders. For example, ANNs are being used to develop tools that can detect cancer in mammograms and skin cancer in images.

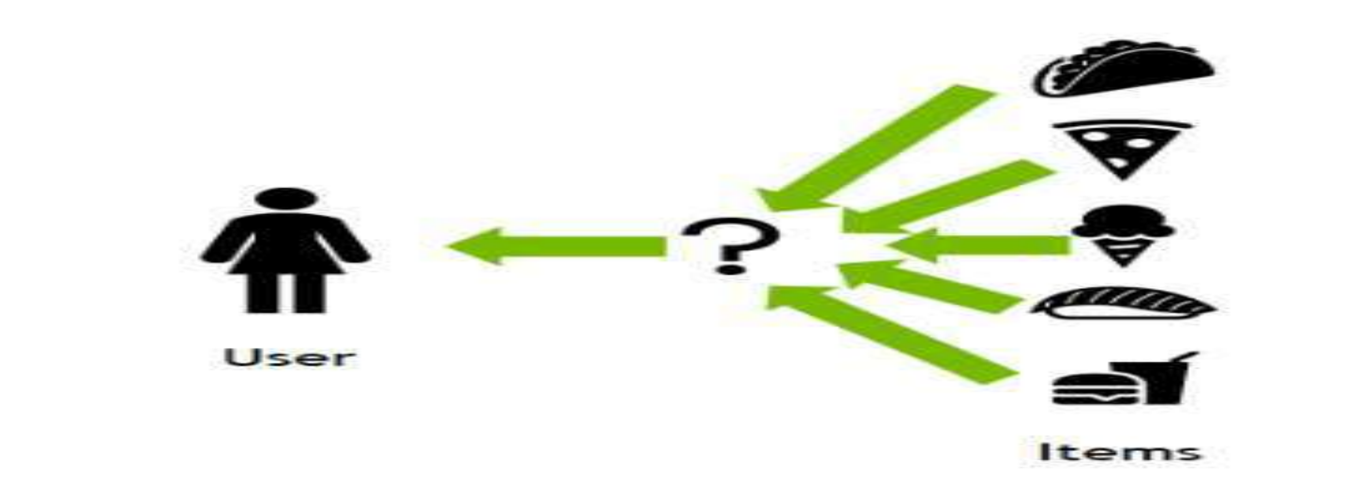
APPLICATIONS OF ANN…..

- **Financial forecasting:** ANNs are used in financial forecasting to predict stock prices, market trends, and creditworthiness. For instance, ANNs are used by some hedge funds to develop trading strategies.

	A B C	E	F	G	H	I	J	K	L
1	© Corporate Finance Institute. All rights reserved.	Historical Results				Forecast Period			
2	Brick 'n' Mortar Co Model	2014	2015	2016	2017	2018	2019	2020	2021
30									
31	Income Statement								
32									
33	**Revenue**	**66,132**	**73,558**	**79,716**	**84,438**	**97,103**	**111,669**	**125,069**	**137,576**
34	Cost of Goods Sold (COGS)	26,884	27,511	29,488	31,760	40,783	45,784	50,028	55,030
35	**Gross Profit**	**39,248**	**46,047**	**50,228**	**52,678**	**56,320**	**65,885**	**75,041**	**82,546**
36	**Expenses**								
37	Marketing, Advertising & Promotion	12,689	13,369	12,882	14,138	15,537	17,867	20,011	22,012
38	General & Administrative	5,670	5,649	6,172	6,391	7,000	7,000	7,000	7,000
39	Depreciation & Amortization	10,165	9,635	9,265	9,006	4,203	4,512	4,760	4,958
70	Interest	1,400	840	840	840	1,344	1,344	1,344	1,344
71	**Total Expenses**	**29,924**	**29,494**	**29,159**	**30,375**	**28,083**	**30,723**	**33,115**	**35,314**
72	**Earnings Before Tax**	**9,324**	**16,554**	**21,069**	**22,303**	**28,237**	**35,161**	**41,927**	**47,232**
73									
74	Taxes	4,858	8,483	10,908	11,598	9,036	11,252	13,417	15,114
75	**Net Earnings**	**4,466**	**8,071**	**10,161**	**10,706**	**19,201**	**23,910**	**28,510**	**32,118**

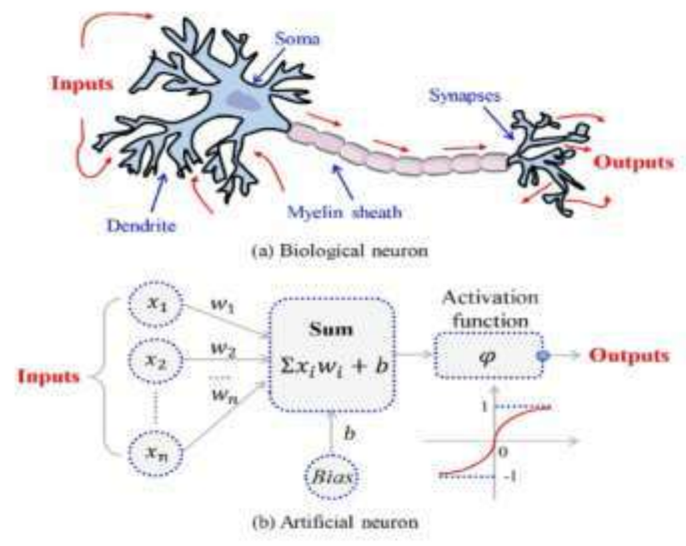
APPLICATIONS OF ANN…..

- **Recommendation systems:** ANNs are used in recommendation systems, which are algorithms that recommend products, services, or content to users. For example, ANNs are used by Amazon and Netflix to recommend products and shows to users.



BIOLOGOCAL NEURON & ARTIFICIAL NEURON:

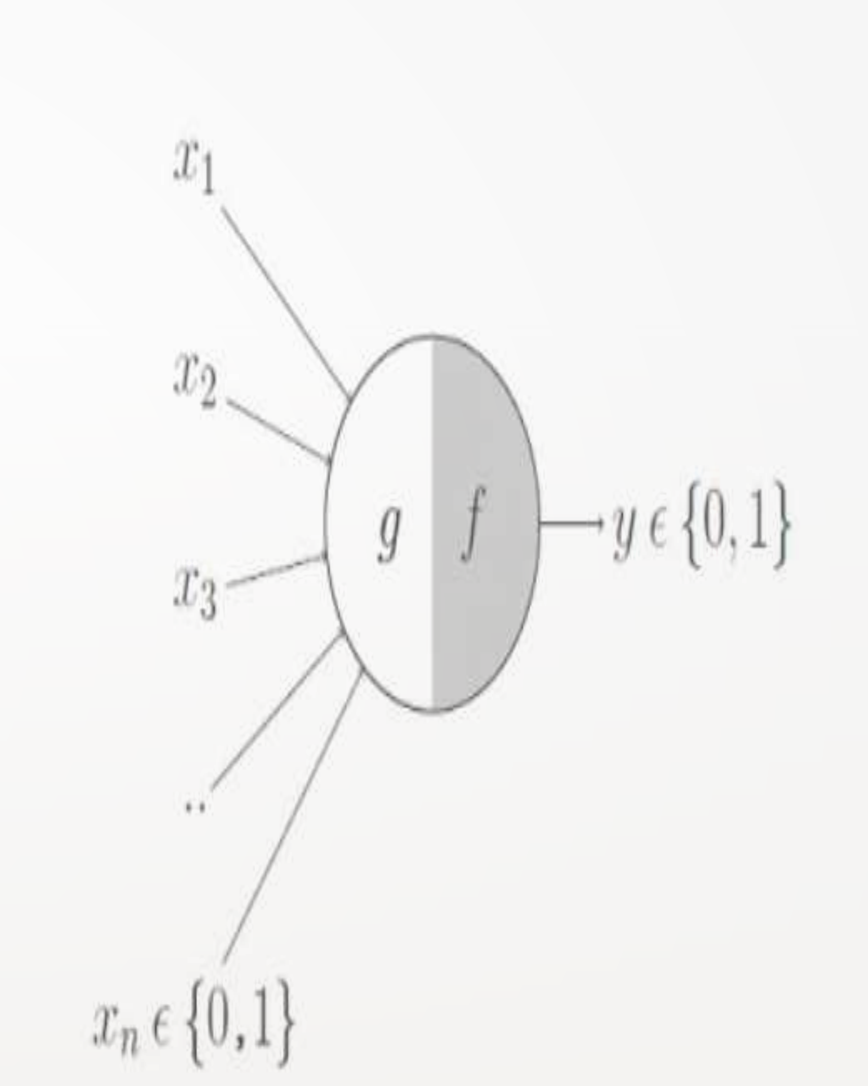
- Biological neurons have a complex and organic structure, consisting of dendrites, soma, axon, and synapses.
- Artificial neurons have a simple and mathematical structure, consisting of inputs, weights, bias, and activation function



Biological Neuron	Artificial Neuron
Dendrite	Inputs
Cell nucleus or Soma	Nodes
Synapses	Weights
Axon	Output

MC-CULLOTH PITTS MODEL

- The first computational model of a neuron was proposed by Warren McCulloch (neuroscientist) and Walter Pitts (logician) in 1943.
- It may be divided into 2 parts. The first part, ***g*** takes an input performs an aggregation and based on the aggregated value the second part, ***f*** makes a decision.



MC-CULLOTH PITTS MODEL-EXAMPLE:

- Lets suppose that I want to predict my own decision, whether to watch a random football game or not on TV. The inputs are all boolean i.e., {0,1} and my output variable is also boolean {0: Will watch it, 1: Won't watch it}.
- So, ***x_1*** could be *isPremierLeagueOn* (I like Premier League more)
- ***x_2*** could be *isItAFriendlyGame* (I tend to care less about the friendlies)
- ***x_3*** could be *isNotHome* (Can't watch it when I'm running errands. Can I?)
- ***x_4*** could be *isManUnitedPlaying* (I am a big Man United fan. GGMU!) and so on.

MC-CULLOTH PITTS MODEL-EXAMPLE:

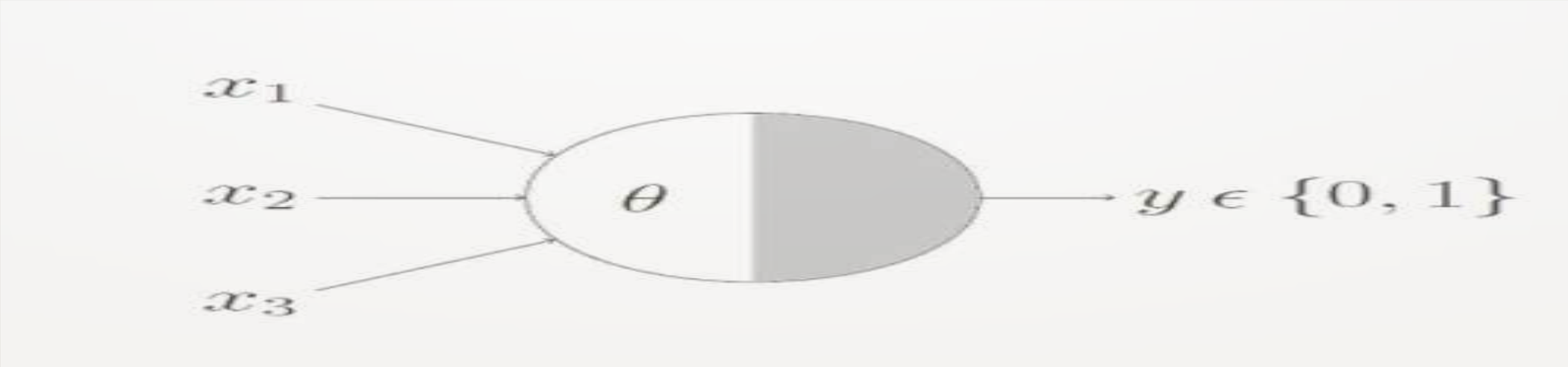
- These inputs can either be *excitatory* or *inhibitory*. Inhibitory inputs are those that have maximum effect on the decision making irrespective of other inputs i.e., if ***x_3*** is 1 (not home) then my output will always be 0 i.e., the neuron will never fire, so ***x_3*** is an inhibitory input.
- Excitatory inputs are NOT the ones that will make the neuron fire on their own but they might fire it when combined together. Formally, this is what is going on:
- We can see that ***g*(x)** is just doing a sum of the inputs — a simple aggregation. And ***theta*** here is called thresholding parameter.
- For example, if I always watch the game when the sum turns out to be 2 or more, the ***theta*** is 2 here. This is called the Thresholding Logic.

$$g(x_1, x_2, x_3, \ldots, x_n) = g(\mathbf{x}) = \sum_{i=1}^{n} x_i$$

$$\begin{aligned} y = f(g(\mathbf{x})) &= 1 \quad if \quad g(\mathbf{x}) \geq \theta \\ &= 0 \quad if \quad g(\mathbf{x}) < \theta \end{aligned}$$

BOOLEAN FUNCTIONS USING M-P NEURON

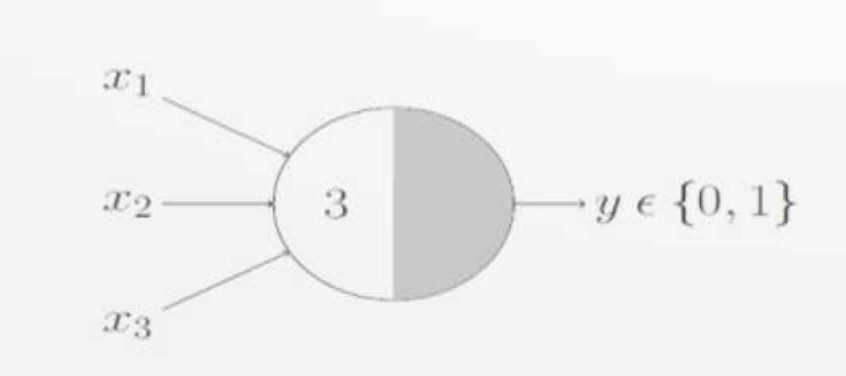
- Now lets look at how this very neuron can be used to represent a few Boolean functions.
- A lot of Boolean decision problems can be cast into this, based on appropriate input variables— like whether to continue reading this PPT, whether to talk to Friends after reading this ppt etc. can be represented by the M-P neuron.
- M-P Neuron: A Concise Représentation just denotes that, for the Boolean inputs ***x_1***, ***x_2*** and ***x_3*** if the ***g*(x)** i.e., **sum ≥ theta**, the neuron will fire otherwise, it won’t.



BOOLEAN FUNCTIONS USING M-P NEURON

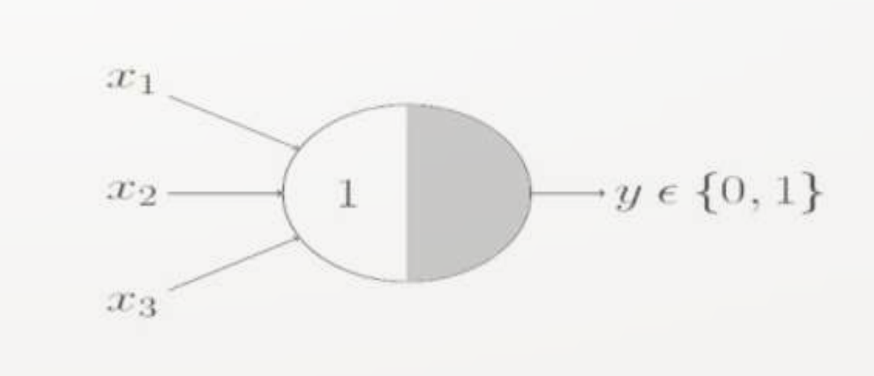
AND GATE

- In AND gate neuron would only fire when ALL the inputs are ON i.e., $\boldsymbol{g}(\mathbf{x}) \geq 3$ here.



OR GATE

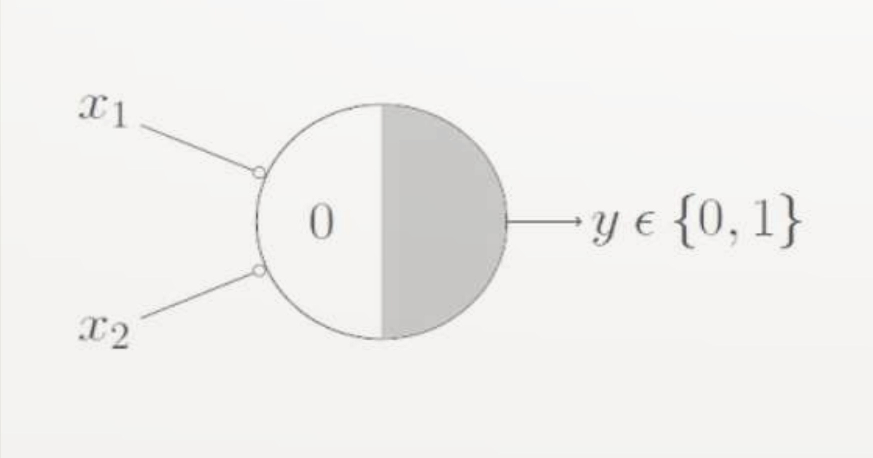
- In OR function neuron would fire if ANY of the inputs is ON i.e., $\boldsymbol{g}(\mathbf{x}) \geq 1$ here.



BOOLEAN FUNCTIONS USING M-P NEURON

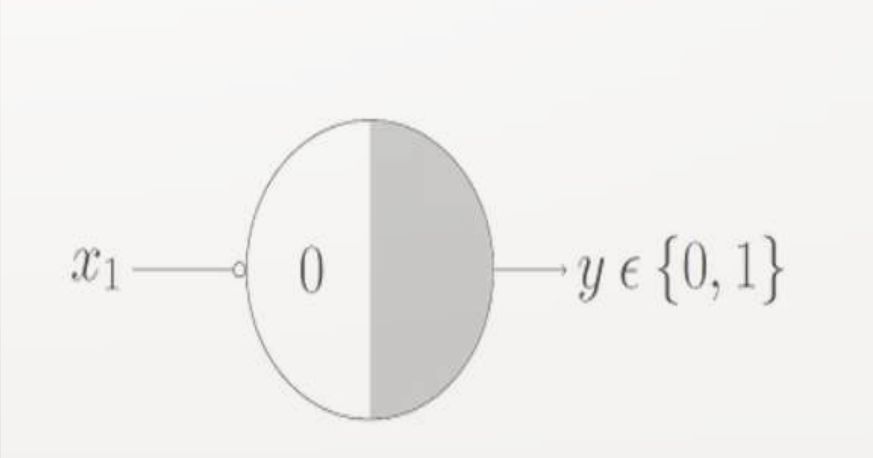
NOR GATE

- For a NOR neuron to fire, we want ALL the inputs to be 0 so the thresholding parameter should also be 0 and we take them all as inhibitory input.



NOT GATE

- For a NOT neuron, 1 outputs 0 and 0 outputs 1. So we take the input as an inhibitory input and set the thresholding parameter to 0.



ADVANTAGES AND LIMITATIONS OF M-P MODEL

ADVANTAGE

- The McCulloch-Pitts neuron is a simplified abstraction of biological neurons and is particularly suitable for modeling simple logical operations.
- It can be used to implement basic logic gates like AND, OR, and NOT.

LIMITATION

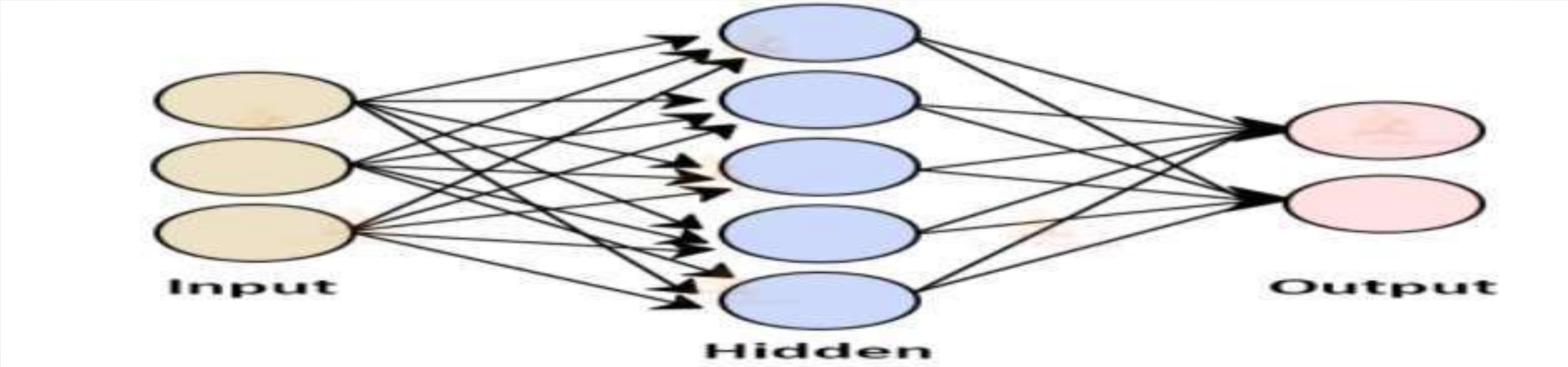
- It has limitations, such as not being able to model complex, continuous functions
- It only allowed for binary inputs and outputs, it only used the threshold step activation function and it did not incorporate weighting the different inputs.

TYPES OF ANN:

- Artificial Neural Networks (ANNs) can be broadly categorized into two main types based on the information flow through the network:
- Feedforward Neural Networks
- Multilayer Perceptrons (MLPs)

TYPES OF ANN:

- **Feedforward Neural Networks:** This is the simplest type of ANN architecture, where the information flows in one direction from input to output. The layers are fully connected, meaning each neuron in a layer is connected to all the neurons in the next layer.

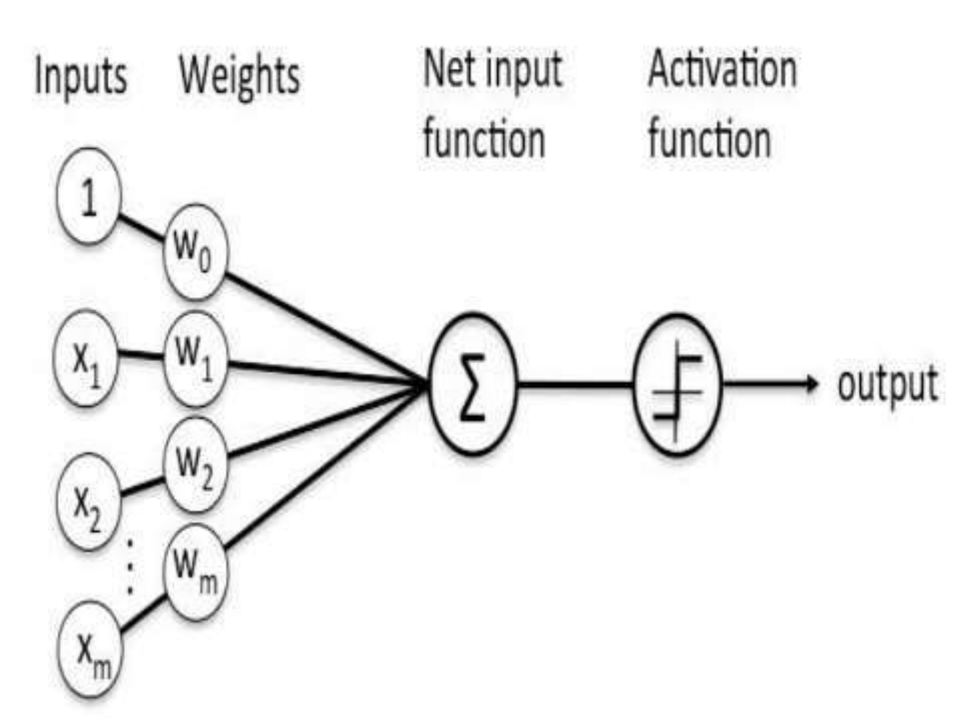


CONTENTS

- **Perceptron model**
- **Single-Layer Perceptron**
- **Multi-Layer Perceptron**
- **Advantages and disadvantages of perceptron**
- **Difference between MC-CULLOTH PITS MODEL AND PERCEPTRON MODEL**

PERCEPTRON MODEL

- Perceptron was introduced by Frank Rosenblatt in 1957.
- He proposed a Perceptron learning rule based on the original MCP neuron.
- A Perceptron is an algorithm for supervised learning of binary classifiers.
- This algorithm enables neurons to learn and processes elements in the training set one at a time.



BASIC COMPONENTS OF PERCEPTRON

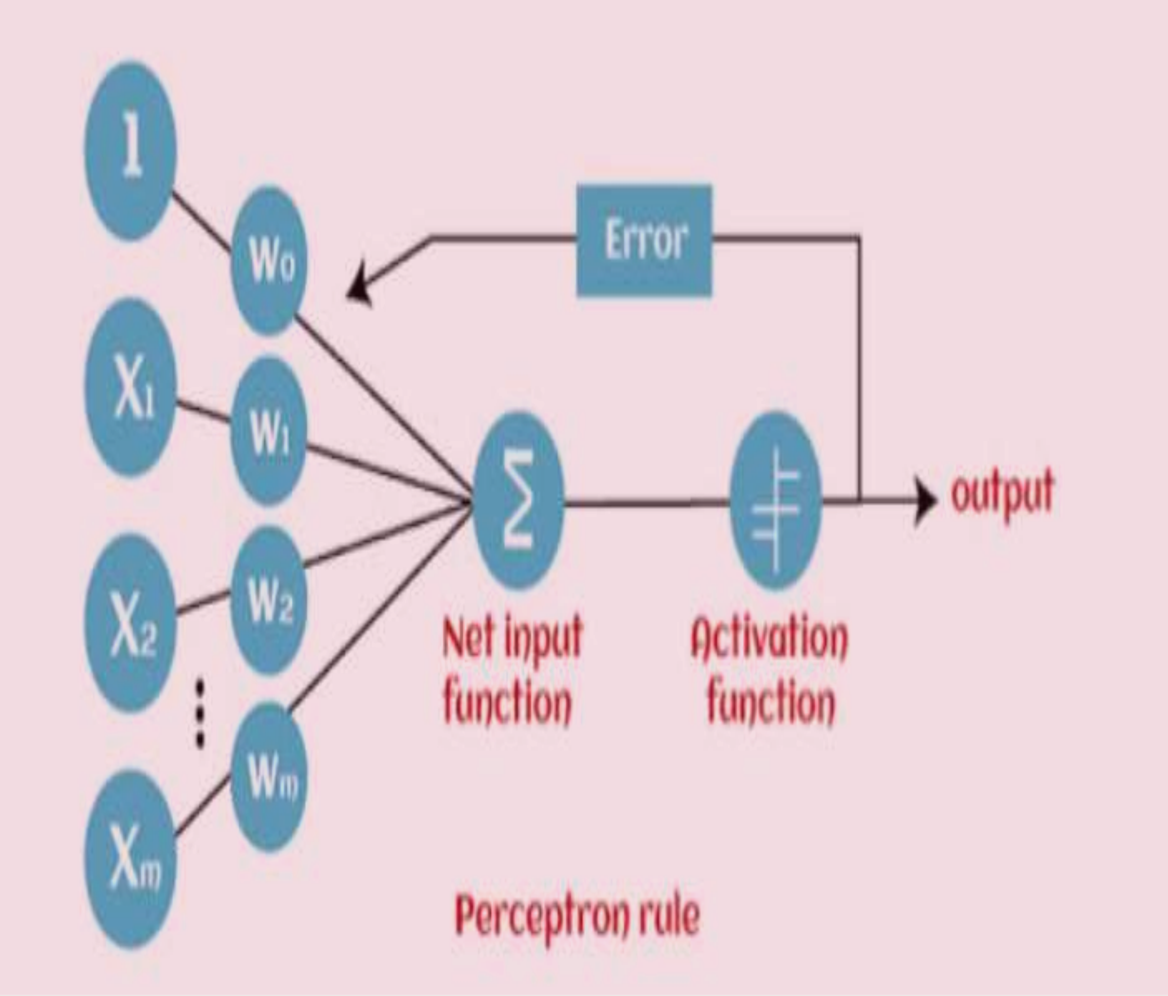
Perceptron is a type of artificial neural network, which is a fundamental concept in machine learning. The basic components of a perceptron are:

1. **Input Layer**: The input layer consists of one or more input neurons, which receive input signals from the external world or from other layers of the neural network.
2. **Weights**: Each input neuron is associated with a weight, which represents the strength of the connection between the input neuron and the output neuron.
3. **Bias**: A bias term is added to the input layer to provide the perceptron with additional flexibility in modeling complex patterns in the input data.
4. **Activation Function**: The activation function determines the output of the perceptron based on the weighted sum of the inputs and the bias term. Common activation functions used in perceptron's include the step function, sigmoid function, and ReLU function.
5. **Output**: The output of the perceptron is a single binary value, either 0 or 1, which indicates the class or category to which the input data belongs.
6. **Training Algorithm**: The perceptron is typically trained using a supervised learning algorithm such as the perceptron learning algorithm or backpropagation. During training, the weights and biases of the perceptron are adjusted to minimize the error between the predicted output and the true output for a given set of training examples.

Overall, the perceptron is a simple yet powerful algorithm that can be used to perform binary classification tasks and has paved the way for more complex neural networks used in deep learning today.

HOW DOES PERCEPTRON WORK?

- Perceptron is considered a single-layer neural link with four main parameters.
- The perceptron model begins with multiplying all input values and their weights, then adds these values to create the weighted sum.
- Further, this weighted sum is applied to the activation function ‘f’ to obtain the desired output.
- This activation function is also known as the step function and is represented by ‘f.’
- This step function or Activation function is vital in ensuring that output is mapped between (0,1) or (-1,1).
- Take note that the weight of input indicates a node’s strength.
- Similarly, an input value gives the ability the shift the activation function curve up or down.



HOW DOES PERCEPTRON WORK?

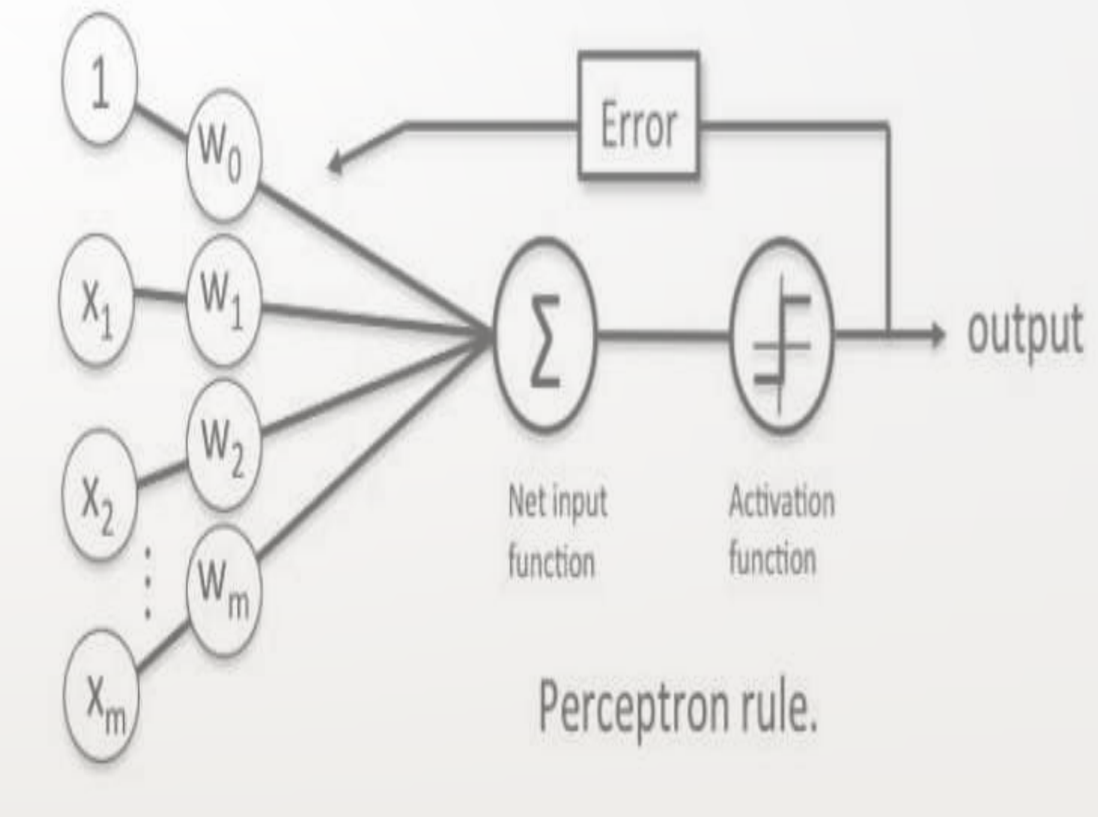
- Step 1: Multiply all input values with corresponding weight values and then add to calculate the weighted sum. The following is the mathematical expression of it:
- ∑wi*xi = x1*w1 + x2*w2 + x3*w3+……..xn*wn
- Add a term called bias ‘b’ to this weighted sum to improve the model’s performance.
- Step 2: An activation function is applied with the above-mentioned weighted sum giving us an output either in binary form or a continuous value as follows:
- Y=f(∑wi*xi + b)

PERCEPTRON LEARNING RULE

- Perceptron Learning Rule states that the algorithm would automatically learn the optimal weight coefficients. The input features are then multiplied with these weights to determine if a neuron fires or not.
- The Perceptron receives multiple input signals, and if the sum of the input signals exceeds a certain threshold, it either outputs a signal or does not return an output.

Error in Perceptron

- In the Perceptron Learning Rule, the predicted output is compared with the known output. If it does not match, the error is propagated backward to allow weight adjustment to happen.



INPUTS OF A PERCEPTRON

- A Perceptron accepts inputs, moderates them with certain weight values, then applies the transformation function to output the final result. The image below shows a Perceptron with a Boolean output.
- A Boolean output is based on inputs such as salaried, married, age, past credit profile, etc. It has only two values: Yes and No or True and False. The summation function "∑" multiplies all inputs of "x" by weights "w" and then adds them up as follows:

$$w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n$$

ACTIVATION FUNCTIONS OF PERCEPTRON

- The activation function applies a step rule (convert the numerical output into +1 or -1) to check if the output of the weighting function is greater than zero or not.
- For example:
- If $\sum$ wixi> 0 => then final output “o” = 1 (issue bank loan)
- Else, final output “o” = -1 (deny bank loan)
- Step function gets triggered above a certain value of the neuron output; else it outputs zero. Sign Function outputs +1 or -1 depending on whether neuron output is greater than zero or not. Sigmoid is the S-curve and outputs a value between 0 and 1.

TYPES OF ACTIVATION FUNCTION

The Activation Functions can be basically divided into 2 types-

1. Linear Activation Function
2. Non-linear Activation Functions

LINEAR ACTIVATION FUNCTION

- As you can see the function is a line or linear. Therefore, the output of the functions will not be confined between any range.
- **Equation :** f(x) = x
- **Range :** (-infinity to infinity)
- It doesn't help with the complexity or various parameters of usual data that is fed to the neural networks.

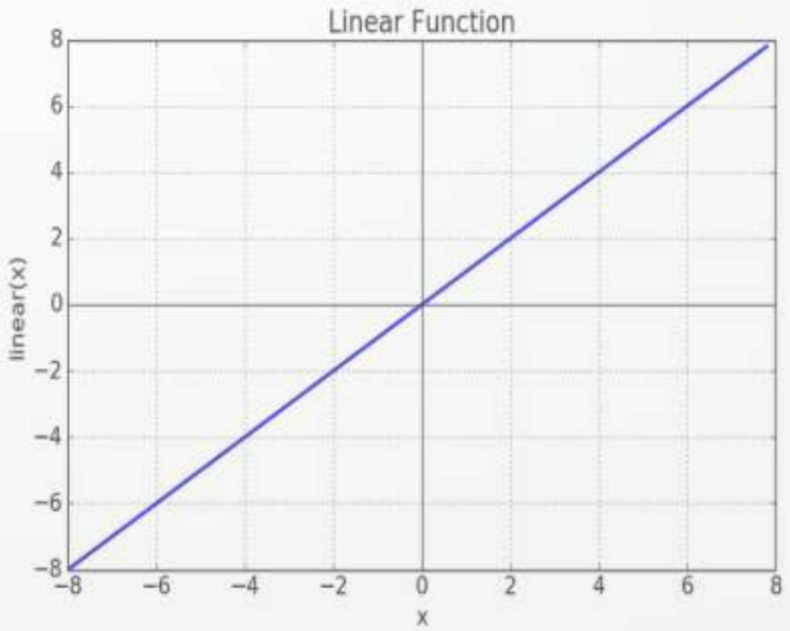


Fig: Linear Activation Function

NON LINEAR ACTIVATION FUNCTION

- The Nonlinear Activation Functions are the most used activation functions. Nonlinearity helps to makes the graph look something like this
- It makes it easy for the model to generalize or adapt with variety of data and to differentiate between the output.
- The main terminologies needed to understand for nonlinear functions are:
- Derivative or Differential: Change in y-axis w.r.t. change in x-axis.It is also known as slope.
- Monotonic function: A function which is either entirely non-increasing or non-decreasing.

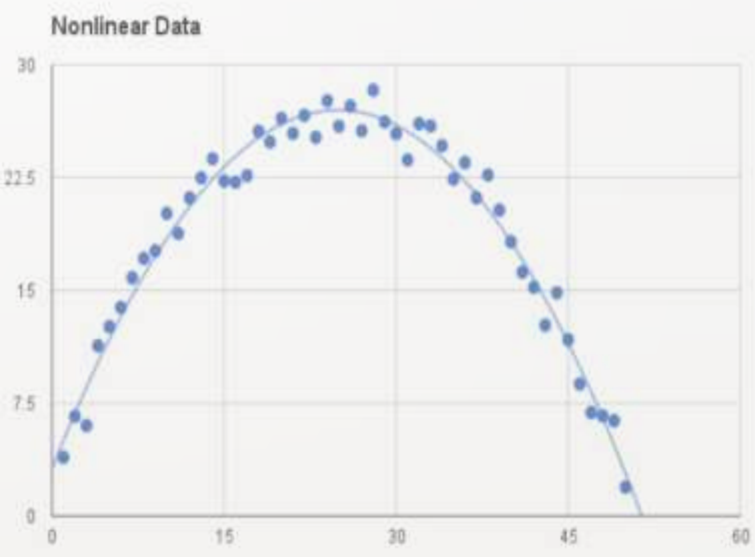
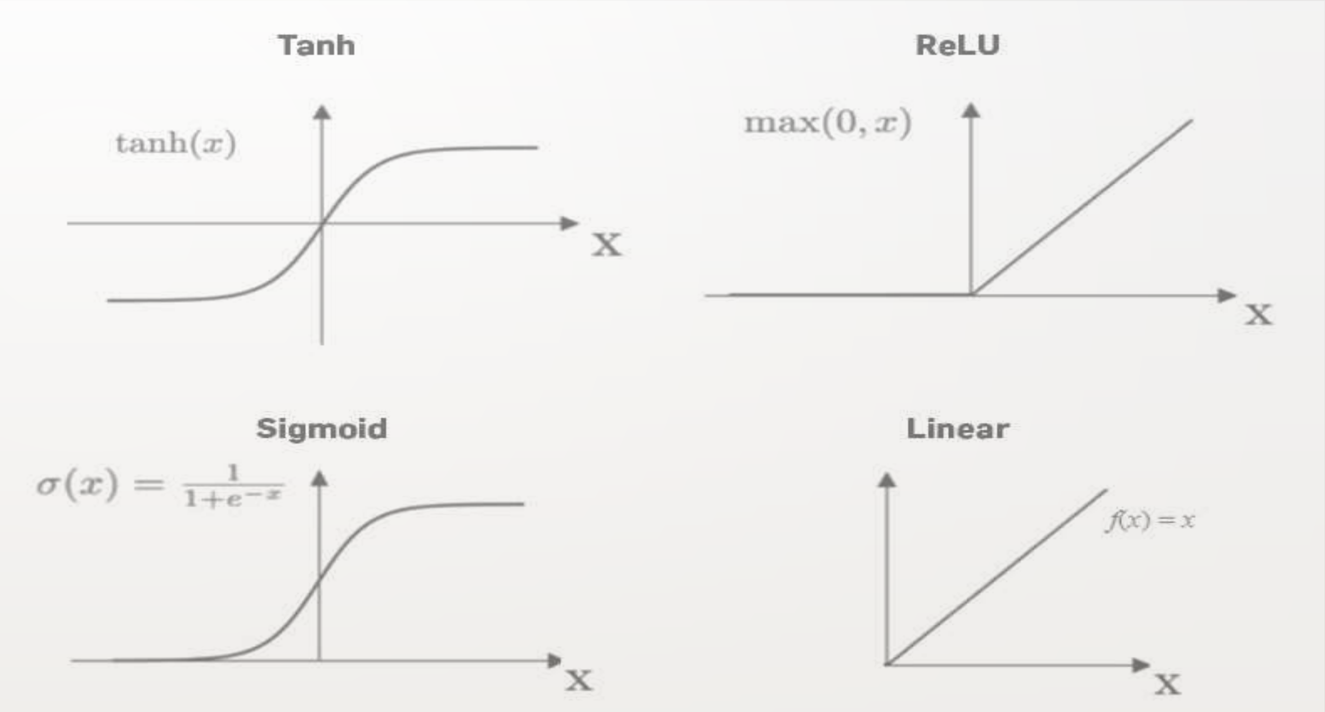


Fig: Non-linear Activation Function

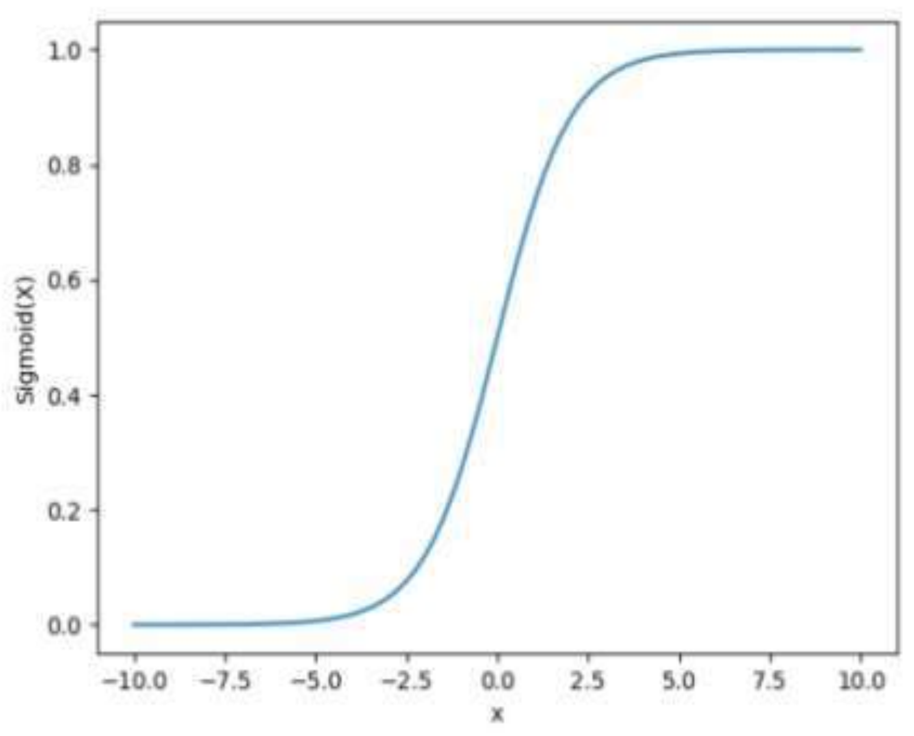
NON LINEAR ACTIVATION FUNCTION…

- The Nonlinear Activation Functions are mainly divided on the basis of their **range or curves-**
- Sigmoid or Logistic Activation Function
- Tanh or hyperbolic tangent Activation Function
- ReLU (Rectified Linear Unit) Activation Function
- Leaky ReLU



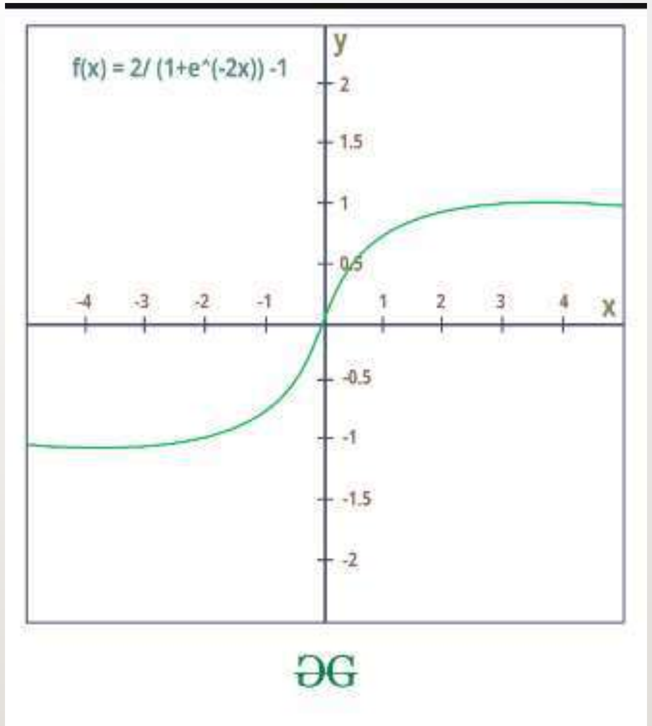
TYPES OF NONLTINEAR ACTIVATION FUNCTION

- **Sigmoid Function**
- It is a function which is plotted as 'S' shaped graph.
- Equation : A = 1/(1 + e-x)
- Nature : Non-linear. Notice that X values lies between -2 to 2, Y values are very steep. This means, small changes in x would also bring about large changes in the value of Y.
- Value Range : 0 to 1
- Uses : Usually used in output layer of a binary classification, where result is either 0 or 1, as value for sigmoid function lies between 0 and 1 only so, result can be predicted easily to be 1 if value is greater than 0.5 and 0 otherwise.



TANH

- The activation that works almost always better than sigmoid function is Tanh function also known as **Tangent Hyperbolic function**. It's actually mathematically shifted version of the sigmoid function. Both are similar and can be derived from each other.
- **Equation :-**
  f(x) = tanh(x) = 2/(1 + e-2x) – 1
  OR
  tanh(x) = 2 * sigmoid(2x) – 1
- **Value Range :-** -1 to +1
- **Nature :-** non-linear
- **Uses :-** Usually used in hidden layers of a neural network as it's values lies between **-1 to 1** hence the mean for the hidden layer comes out be 0 or very close to it, hence helps in *centering the data* by bringing mean close to 0. This makes learning for the next layer much easier.



RELU

- It Stands for *Rectified linear unit*. It is the most widely used activation function. Chiefly implemented in *hidden layers* of Neural network.
- **Equation :-** ***A(x) = max(0,x)***. It gives an output x if x is positive and 0 otherwise.
- **Value Range :-** [0, inf)
- **Nature :-** non-linear, which means we can easily backpropagate the errors and have multiple layers of neurons being activated by the ReLU function.
- **Uses :-** ReLu is less computationally expensive than tanh and sigmoid because it involves simpler mathematical operations. At a time only a few neurons are activated making the network sparse making it efficient and easy for computation.
- In simple words, RELU learns *much faster* than sigmoid and Tanh function.

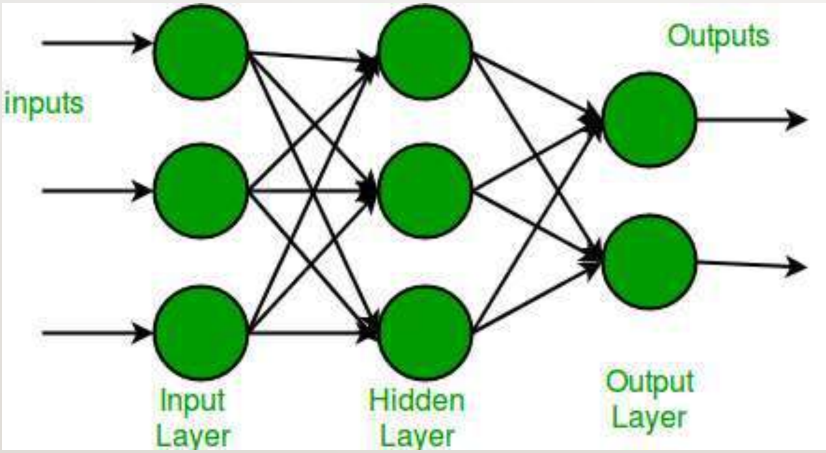
OUTPUT OF PERCEPTRON

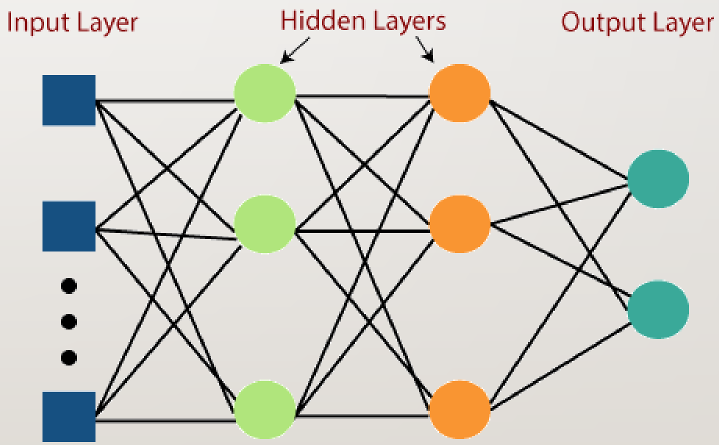
- Perceptron with a Boolean output:
- Inputs: x1…xn
- Output: o(x1….xn)
- Weights: wi=> contribution of input xi to the Perceptron output;
- w0=> bias or threshold
- If ∑w.x > 0, output is +1, else -1. The neuron gets triggered only when weighted input reaches a certain threshold value.

$$o(x_1, \ldots, x_n) = \begin{cases} 1 \text{ if } w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n > 0 \\ -1 \text{ otherwise} \end{cases}$$

TYPES OF PERCEPTRON

- Single Layer Perceptron model
- Multi-Layered Perceptron model
- **Single Layer Perceptron model**: One of the easiest ANN(Artificial Neural Networks) types consists of a feed-forward network and includes a threshold transfer inside the model.
- **Multi-Layered Perceptron model**: It is mainly similar to a single-layer perceptron model but has more hidden layers.





ADVANTAGES AND DISADVANTAGES OF PERCEPTRON

ADVANTAGES

1. In Perceptron, the weight coefficient is automatically learned.
2. Initially, weights are multiplied with input features, and then the decision is made whether the neuron is fired or not.
3. The activation function applies a step rule to check whether the function is more significant than zero.

DISADVANTAGES

1. The output of a perceptron can only be a binary number (0 or 1) due to the hard-edge transfer function.
2. It can only be used to classify the linearly separable sets of input vectors. If the input vectors are non-linear, it is not easy to classify them correctly

DIFFERENCE BETWEEN MC-CULLOTH PITS MODEL AND PERCEPTRON MODEL

MC-CULLOCH PITS MODEL

- McCulloh/Pitt's Model accepts only boolean inputs
- Both, MP Neuron Model as well as the Perceptron model work on linearly separable data.
- In McCulloh/Pitt's Model the inputs are not weighted which means that this model is not flexible.

PERCEPTRON MODEL

- Perceptron Model can process inputs in various real forms.
- Both, MP Neuron Model as well as the Perceptron model work on linearly separable data.
- Perceptron model accepts weights with respect to the provided inputs which makes it much more flexible.

WEB REFERENCES

[1] https://www.guru99.com/backpropogation-neural-network.html

[2] https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/

[3] http://neuralnetworksanddeeplearning.com/chap2.html

Conclusion

- Artificial Neural Networks are an imitation of the biological neural networks, but much simpler ones.
- The computing would have a lot to gain from neural networks. Their ability to learn by example makes them very flexible and powerful furthermore there is need to device an algorithm in order to perform a specific task.

REFERENCES

- Craig Heller, and David Sadava, *Life: The Science of Biology, fifth edition*, Sinauer Associates, INC, USA, 1998.
- Introduction to Artificial Neural Networks, Nicolas Galoppo von Borries
- Tom M. Mitchell, *Machine Learning, WCB McGraw-Hill, Boston, 1997.*

TERMINAL QUESTIONS

- What is an Artificial Neural Network (ANN), and how does it work?
- What are the key components of an ANN?
- What are some common applications of ANNs in various industries?
- What is the role of activation functions in ANNs, and what are some commonly used activation functions?
- What is the McCulloch-Pitts neuron model, and how does it function?
- What is the significance of the threshold and binary output in the McCulloch-Pitts model?
- How can the McCulloch-Pitts model be used to represent basic logical operations (e.g., AND, OR, NOT)?
- What are the key components of the Perceptron model (e.g., weights, bias, activation function)?

Self-Assessment Questions

1. Single layer perceptron can learn ____________.

(a) To mimic the human brain

(b) To process large datasets

(c) To perform linear regression

(d) To solve differential equations

Answer: A. To mimic the human brain

2. Which of the following is NOT a common activation function used in ANNs?

(a) Sigmoid

(b) ReLU

(c) Hyperbolic tangent

(d) Linear regression

Answer: D. Linear regression

Self-Assessment Questions

1. The McCulloch-Pitts model is primarily designed to:

(a) Perform arithmetic calculations
(b) Simulate biological neurons
(c) Analyze economic data
(d) Classify non-linear data

Answer: B. Simulate biological neurons

2. In the McCulloch-Pitts model, logical operations such as AND, OR, and NOT can be represented by adjusting:

(a) Activation functions
(b) Input weights and thresholds
(c) Learning rates
(d) Number of neurons

Answer: B. Input weights and thresholds

Self-Assessment Questions

1. Who developed the Perceptron model?

(a) Geoffrey Hinton

(b) Yann LeCun

(c) Frank Rosenblatt

(d) Andrew Ng

Answer: C. Frank Rosenblatt

2. The Perceptron algorithm updates its weights based on:

(a) The gradient of the error

(b) The difference between the expected and actual output

(c) The average of all inputs

(d) The sum of squared errors

Answer: B. The difference between the expected and actual output

THANK YOU