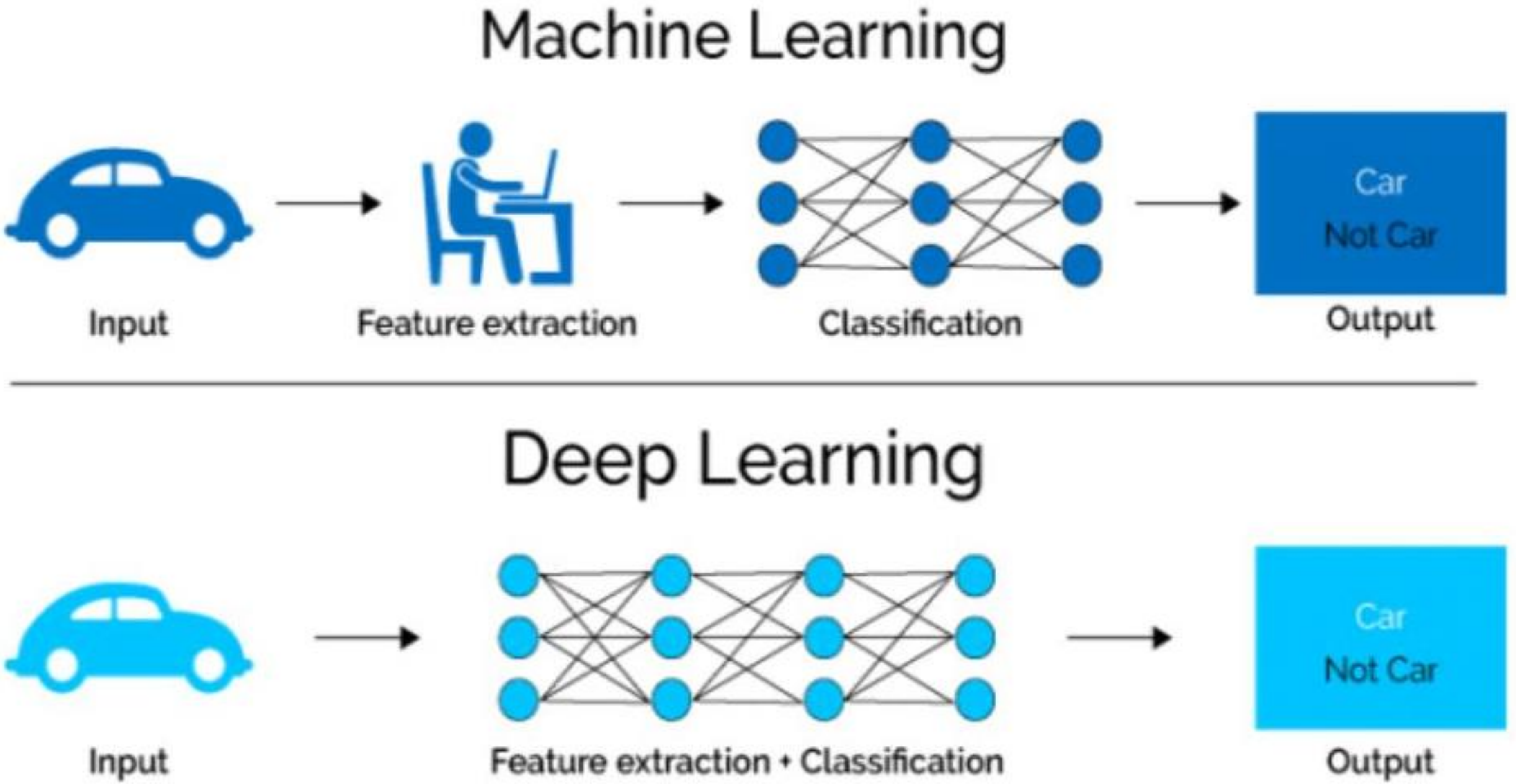
Introduction
to
Deep Learning
and
Artificial Neural Networks

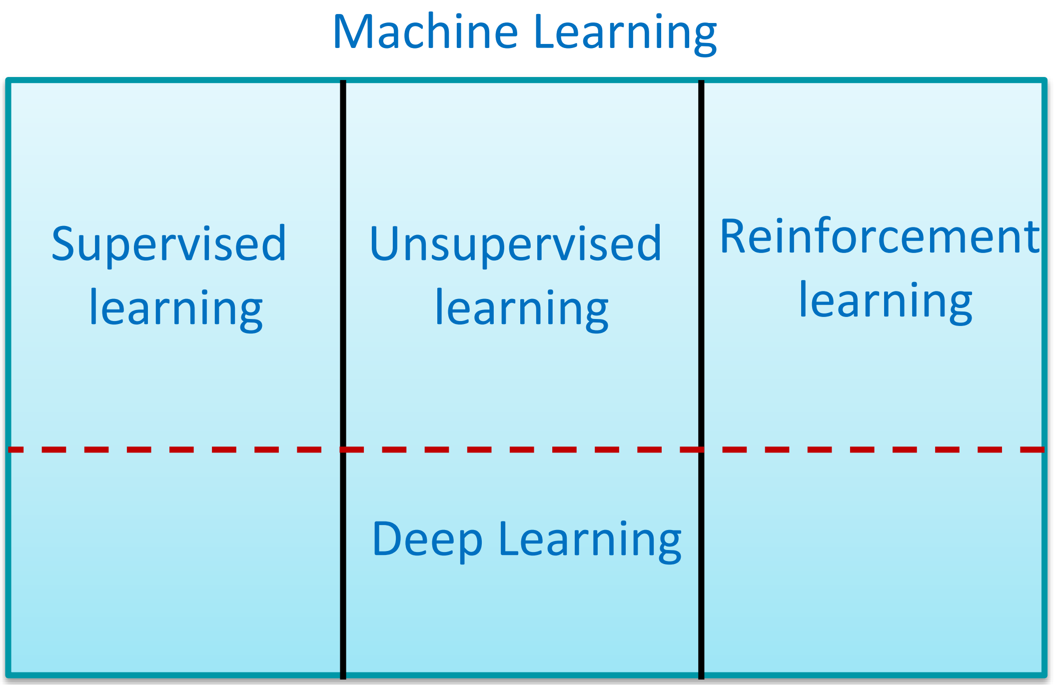
Contents

Introduction to Deep Learning (DL)


Machine Learning
Car
Not Car
Input
Feature extraction
Classification
Output
Deep Learning
Car
Not Car
Input
Feature extraction + Classification
Output

Intro to Deep Learning

- Sub-field of Machine Learning
- Algorithms are developed inspired by the structure and functioning of human brain
- Uses artificial neural networks to learn and perform tasks
- Basic unit of a neural network is a neuron whose functioning is similar to a biological neuron

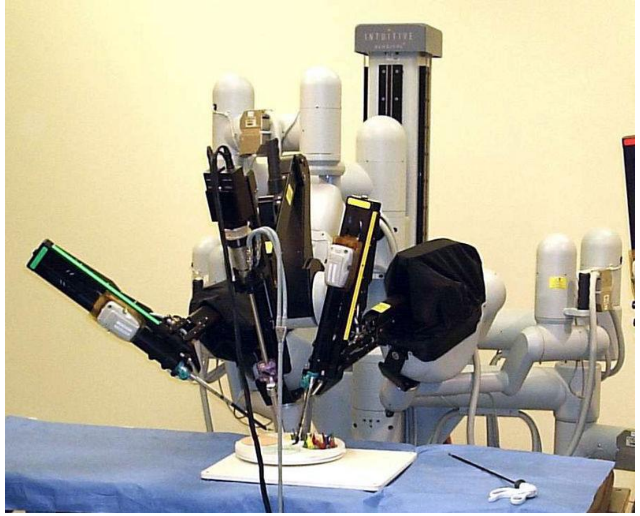


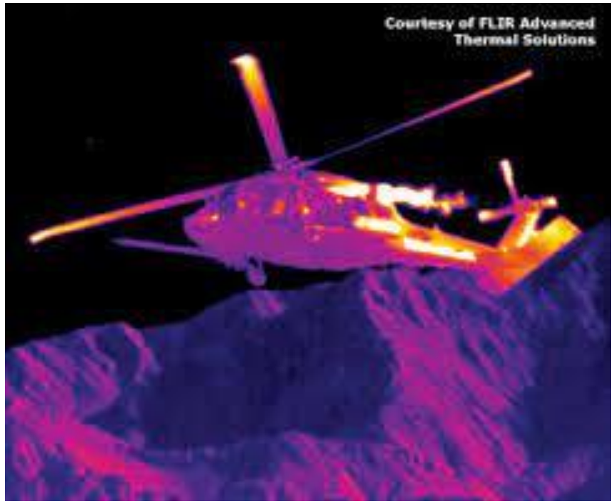
Applications of Deep Learning

- Computer Vision based Applications
  - To extract information from images and videos
    - Recognise objects
    - Gauge properties of objects
    - Predict what the objects are doing
  - To deduce, plan and reason based on extracted information
- Natural Language Processing (NLP) based Applications
  - To process, analyse and interpret text or natural language data
- Other Applications
  - To make predictions based on non-image and non-text data

Applications of DL in Computer Vision

- Medical Imaging
  - Patient diagnosis
  - Medical surgeries
- Object detection
  - Traffic management and control
  - Defense strategies
  - Self driving cars

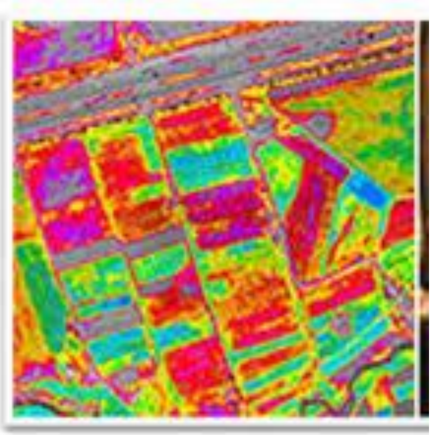




Images Source: Wikipedia/ Google Images

Applications of DL in Computer Vision

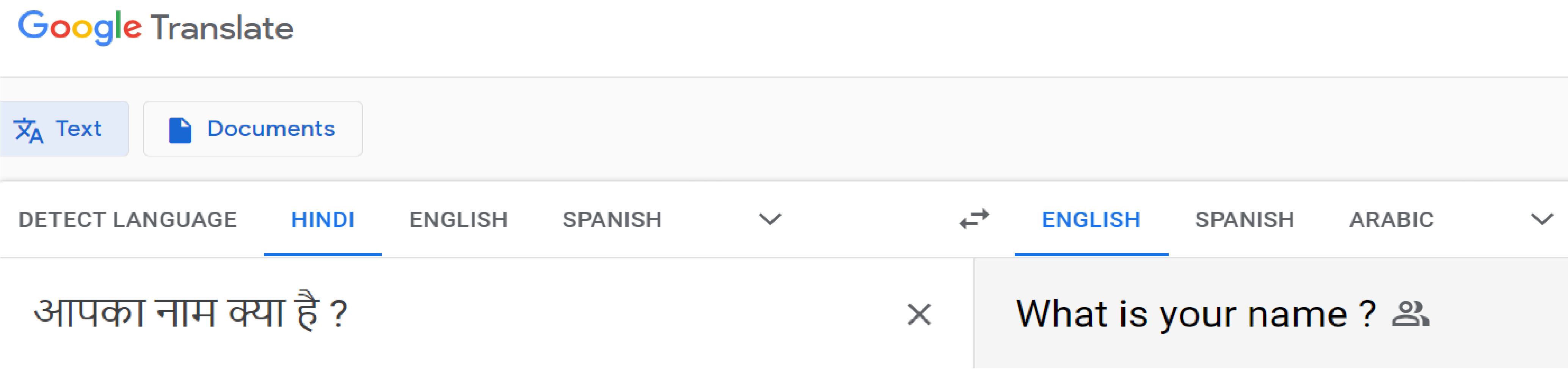
- Face & Fingerprint Recognition
  - Social media platforms
  - Security applications
- Special effects in movies and broadcast of sporting events
- Remote sensing applications
  - Navigational purpose
  - Agriculture, Research, MARS, MOON



Images Source: Wikipedia/ Google Images

Applications of DL in NLP

- Spam Detection:
  - ✓ Scanning emails for words that indicate spam
- Machine Translation:
  - ✓ Google translate is the best example of machine translation
  - ✓ Capturing the meaning and tone of the source language is important



Applications of DL in NLP

- **Virtual Agents and Chat Bots:**
  - ✓ Siri and Alexa are examples of virtual agents that can take voice commands and perform tasks
  - ✓ Chat bots are developed to respond to human typed questions with helpful answers
  - ✓ Most websites which directly interact with many consumers have these chat boxes
- **Social Media Sentiment Analysis:**
  - ✓ Analysing social media posts, reviews, etc. to extract response (positive/negative) to products, events, movies, etc.
- **Text Summarisation:**
  - ✓ To ingest huge volumes of text and create summaries for indexes, busy readers, etc.

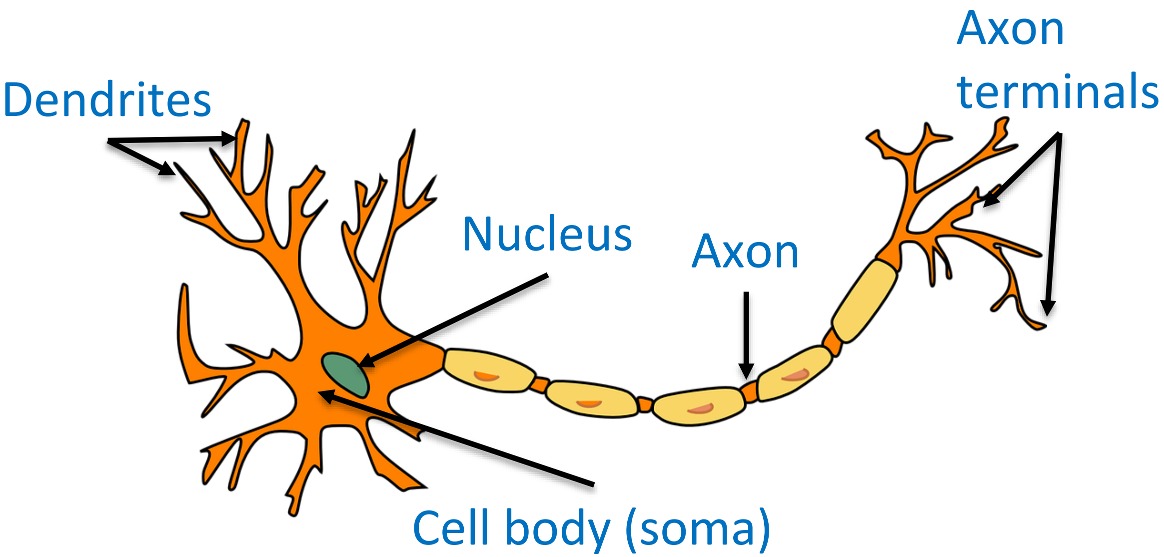
Other Applications of DL

- Healthcare Analytics
  - ✓ Analysing laboratory results to predict the diagnoses and even possible medication
  - ✓ Using lifestyle information provided by smartphones and wearable devices can be used to monitor medical risks
- Fraud Detection: Detecting frauds in financial transactions – mainly credit card frauds
- Recommender systems: Recommending products or movies to consumers based on their historical consumption
- Targeted Advertisements: Getting insights into customer behaviour and needs and targeting ads accordingly
- Predictive Maintenance: To predictively maintain the machine parts in an industries based on certain parameter capturing their wear and tear

Motivation: Biological Neuron to Artificial Neuron

Biological Neuron

- Basic working unit of the brain and nervous system
- Close to 100 billion interconnected neurons in a human brain
- Function together to aid decision making
- Parts and Functioning:
  - Dendrite: Takes signals (stimulus) from the other neurons or other cells in the body
  - Cell body (soma): Processes the signal and may or may not fire the neuron – excitation and inhibition
  - Axon: Transmits the output (response) to other neurons or cells

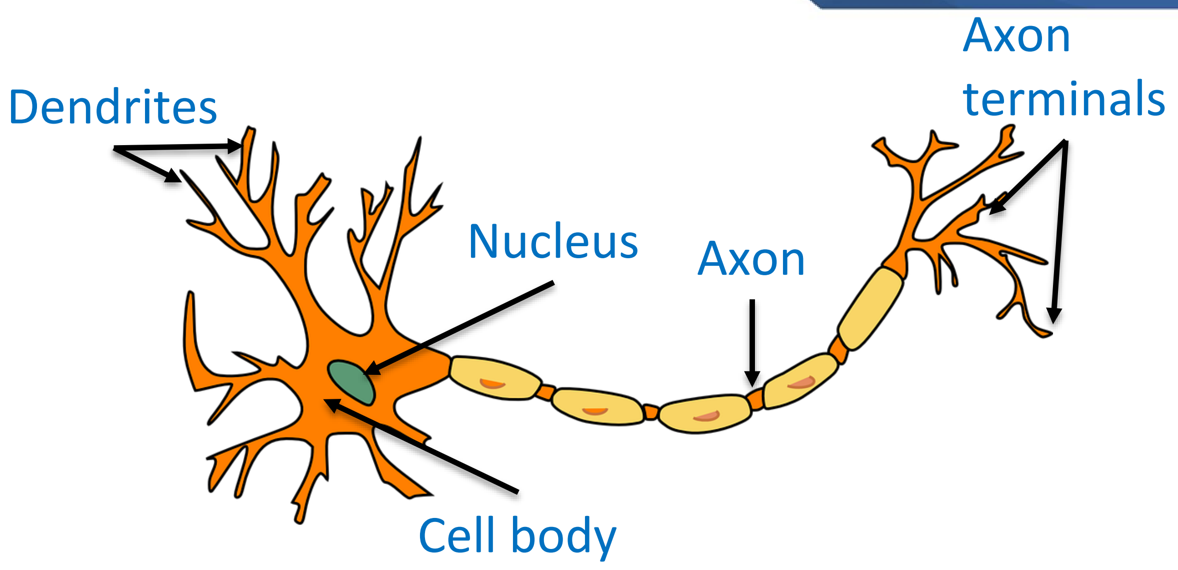


Biological Neuron

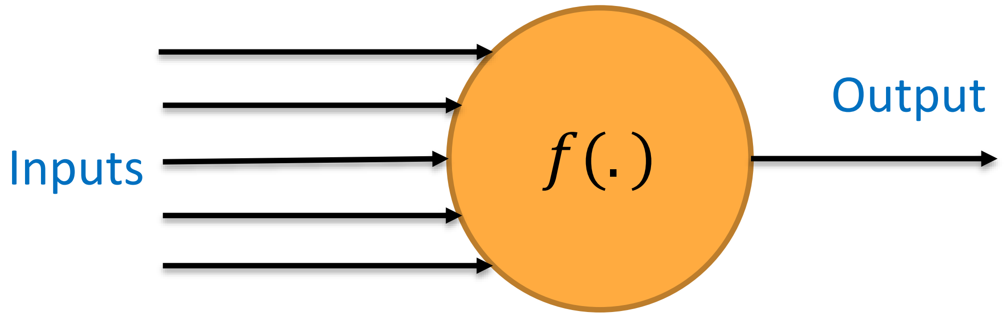
Biological Neuron to Artificial Neuron

Artificial neuron

- Mathematical model of a biological neuron
- Mimics the functioning of a biological neuron
- Takes input in the form of numbers
- Processes the input to give out an output
- Output = f(inputs)
- Different models of artificial neurons have been developed based on this idea



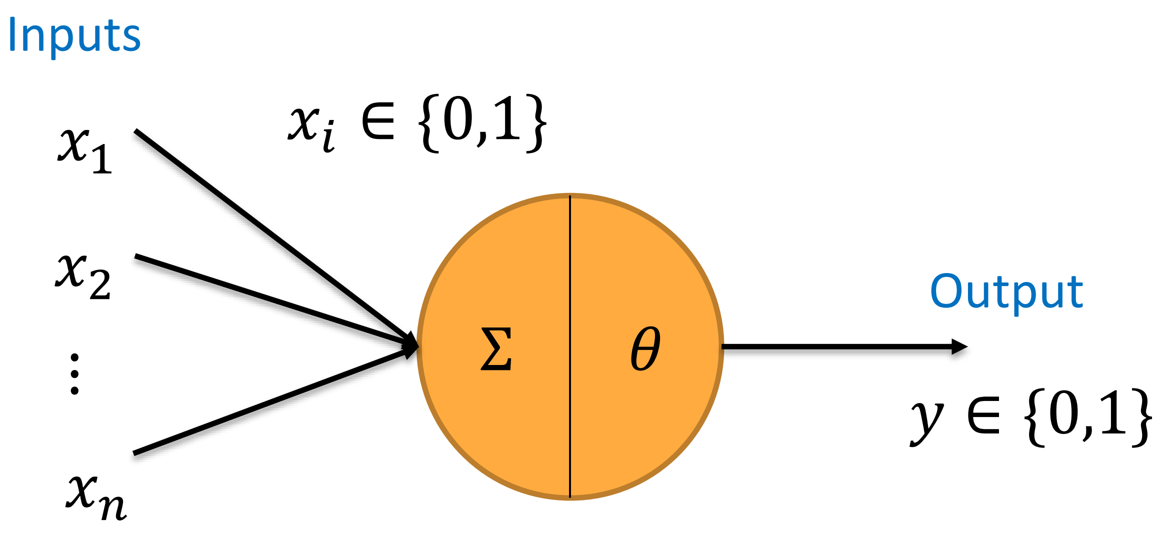
Biological Neuron



Artificial Neuron

Artificial Neuron: McCulloch Pitts Model

- Inputs $(x_1, x_2, \dots, x_n)$ are binary numbers (0 or 1)
- Input can be excitatory or inhibitory
- If any input is inhibitory, then output is zero
- Aggregated excitatory input passes through an activation function to give output (y)
- Activation function is based on thresholding logic
- Here, model refers to the function relating the output to inputs



McCulloch Pitts Model

$$a = x_1 + x_2 + \cdots + x_n$$

$$y = f(a) = 1 \text{ if } a \geq \theta$$

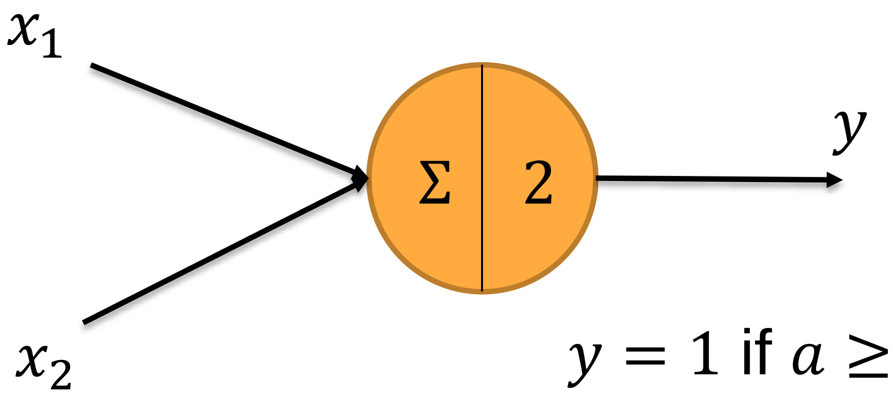
$$y = f(a) = 0 \text{ if } a < \theta$$

McCulloch Pitts Model: Boolean Functions

- This model can be used to represent most Boolean functions

Logical And Function (2 inputs)

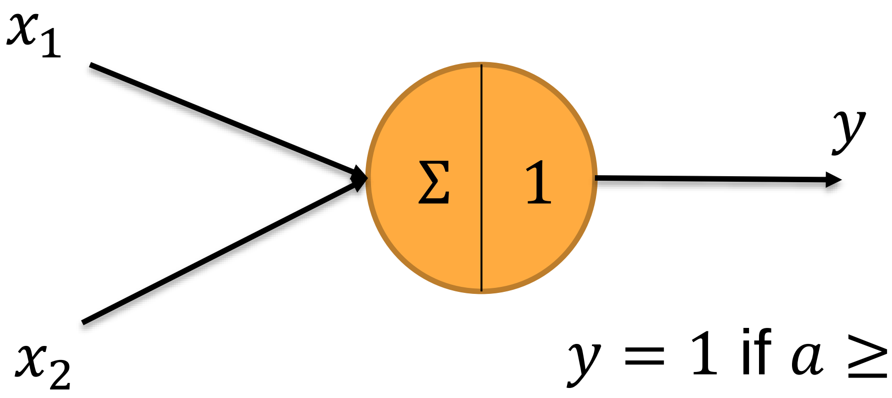
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1



$y = 1$ if $a \geq 2$
$y = 0$ if $a < 2$

Logical Or Function (2 inputs)

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



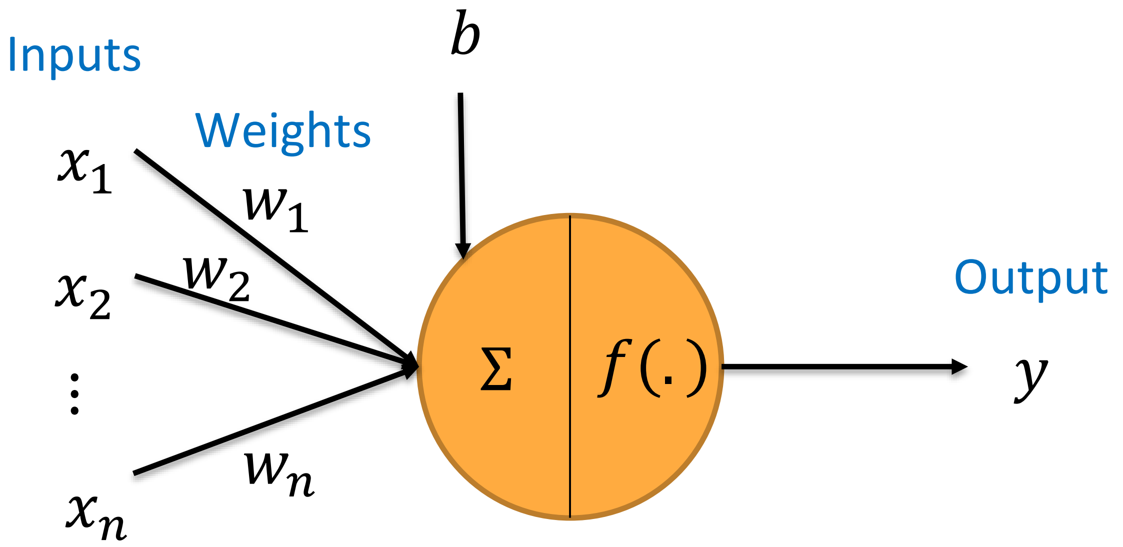
$y = 1$ if $a \geq 1$
$y = 0$ if $a < 1$

McCulloch Pitts Model: Drawbacks

- Drawbacks:
  - Cannot handle non-boolean inputs and outputs.
  - Deciding a appropriate threshold value might be hard as the number of inputs increases.
  - Equal weightage to all inputs – What if more importance is to be attached to some inputs?
- How to overcome these issues? – Perceptron Model

Artificial Neuron: Perceptron Model

- Inputs $(x_1, x_2, \ldots, x_n)$ are real numbers
- Neuron takes the weighted combination of inputs
- Bias (b) is added to weighted inputs
- Weighted input passes through an activation function to give output (y)



Perceptron Model

$$a = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$$

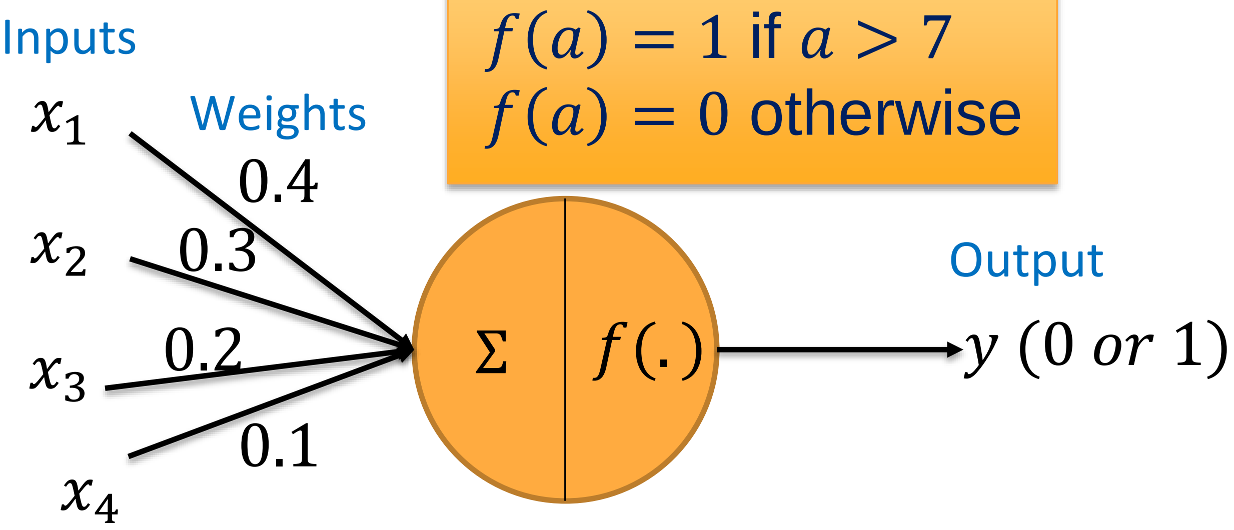
$$y = f(a) = f(w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + \mathrm{b})$$

Artificial Neuron: A Simple Example

- Using an artificial neuron to decide whether to watch a movie or not

Feature	Value for Movie 1	Value for Movie 2
Lead Actor (x_1)	10	7
Director (x_2)	8	5
Thrill factor (x_3)	8	9
Run time (x_4)	9	5

$f(a) = 1$ if $a > 7$
$f(a) = 0$ otherwise



Perceptron Model

Movie 1

$a = 8.9$

$y = f(a) = 1$

Movie 2

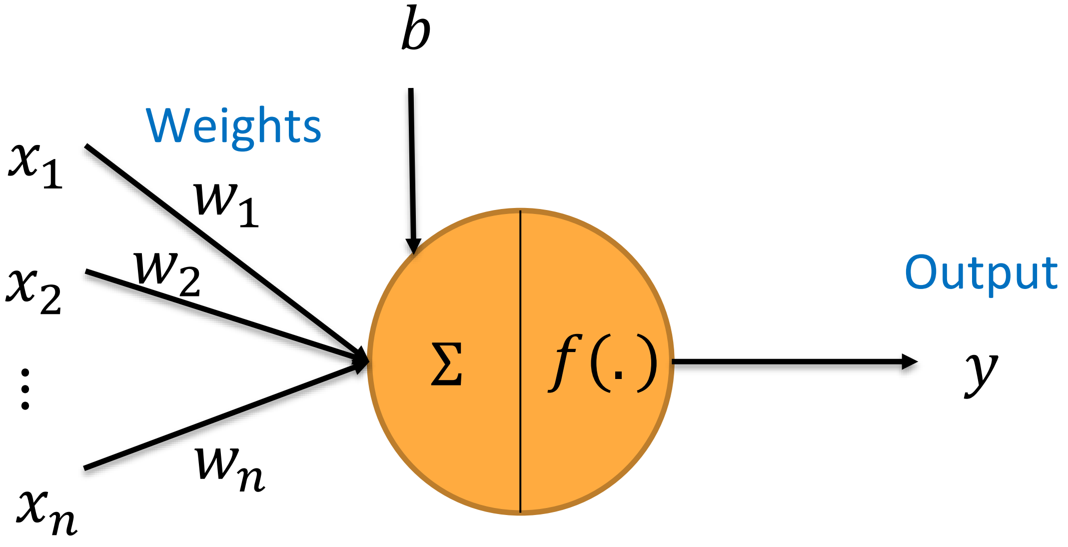
$a = 6.7$

$y = f(a) = 0$

Artificial Neural Network

Artificial Neural Network: Motivation

- One neuron is not sufficient to take complex decisions (complex functions)
- Again, inspired by brain neural network, artificial neural network was developed
- In the brain, many neurons are involved in taking a decision
- All the neurons are inter-connected in the brain
- They are arranged hierarchically in layers



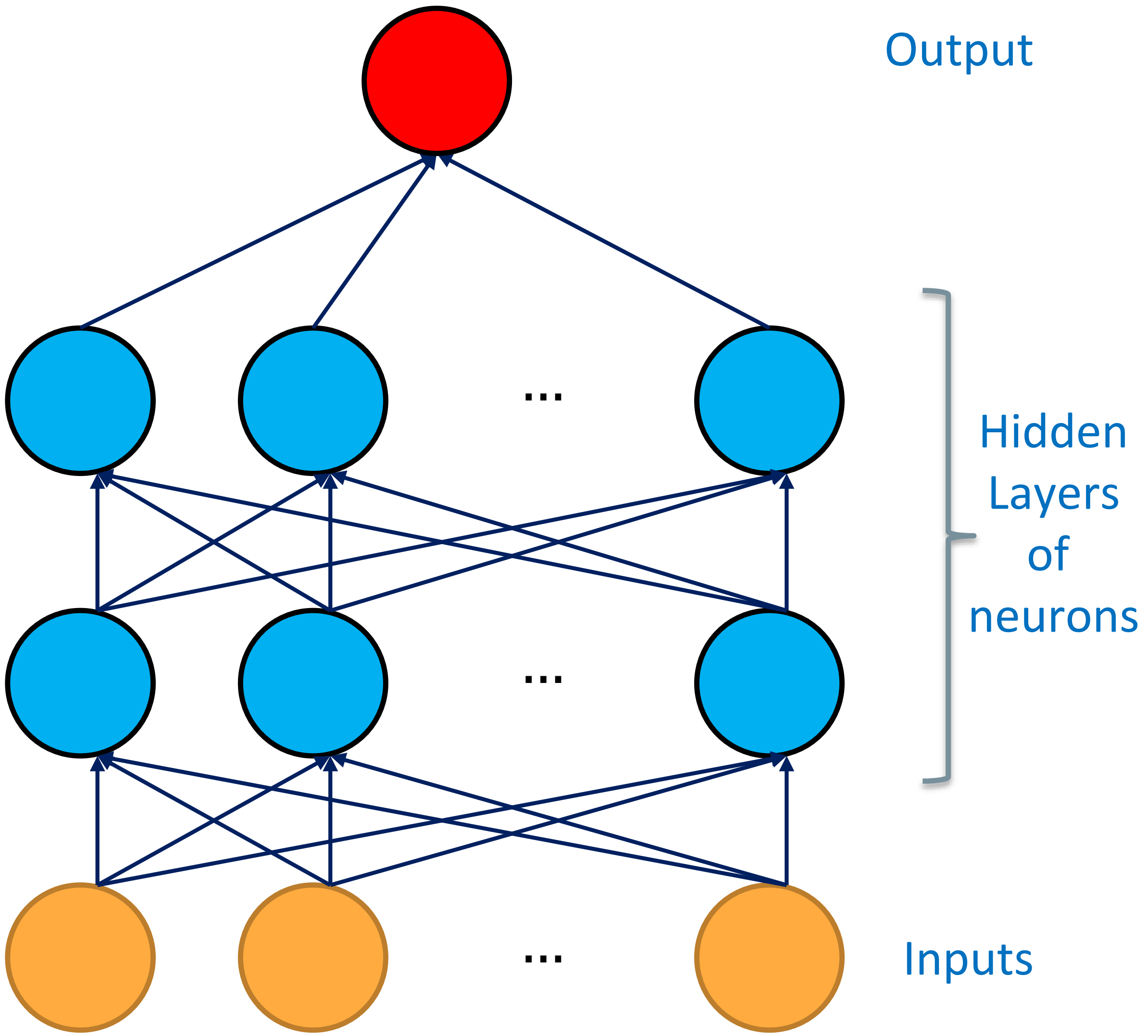
Perceptron Model

$$a = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$$

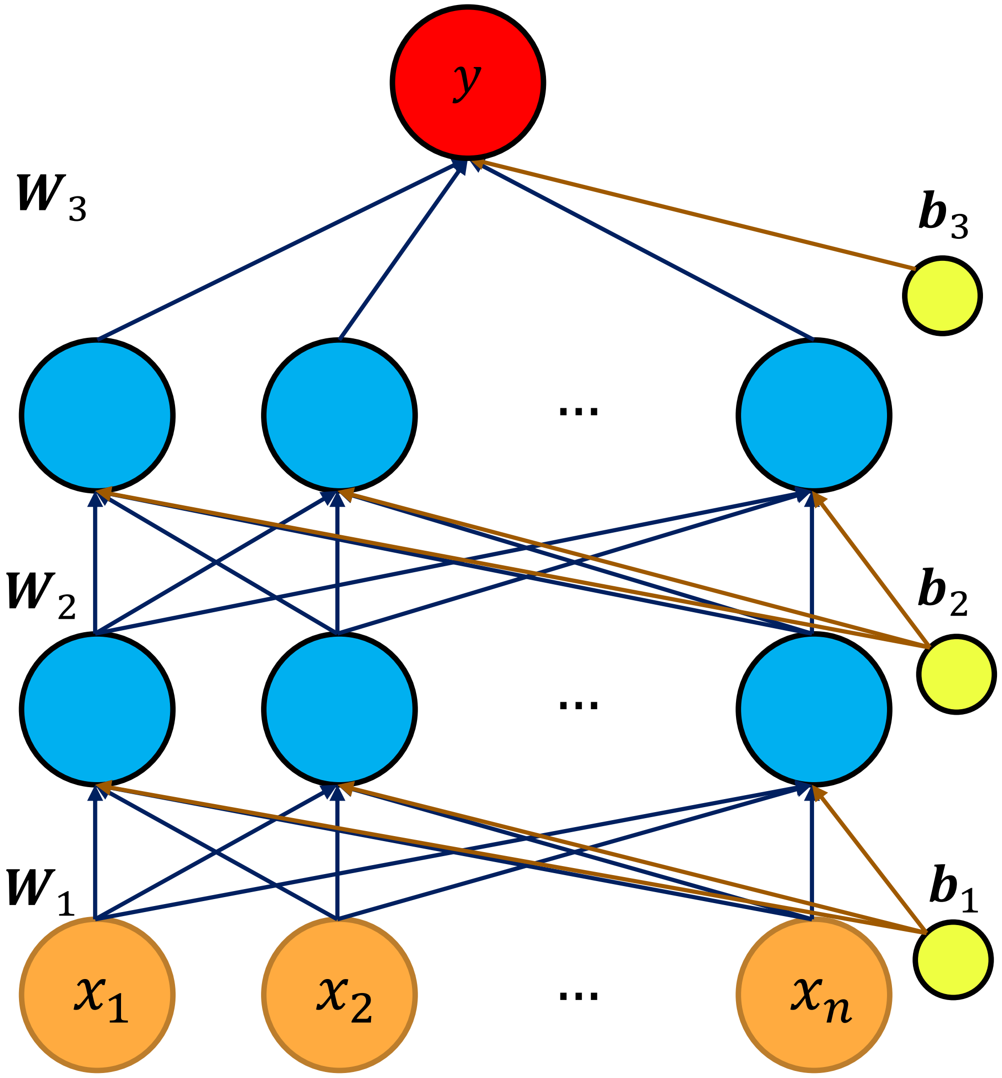
$$y = f(a) = f(w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + \mathrm{b})$$

Artificial Neural Network (ANN)

- ANN consists of multiple layers with multiple neurons in each layer (hidden layers)
- Each neuron (except inputs) represent a perceptron model
- Every neuron in one layer is connected to every neuron in the successive layer
- Output of one neuron are passed as inputs to the neurons of next layer



ANN Architecture



- Input layer (0^{th}) with n inputs

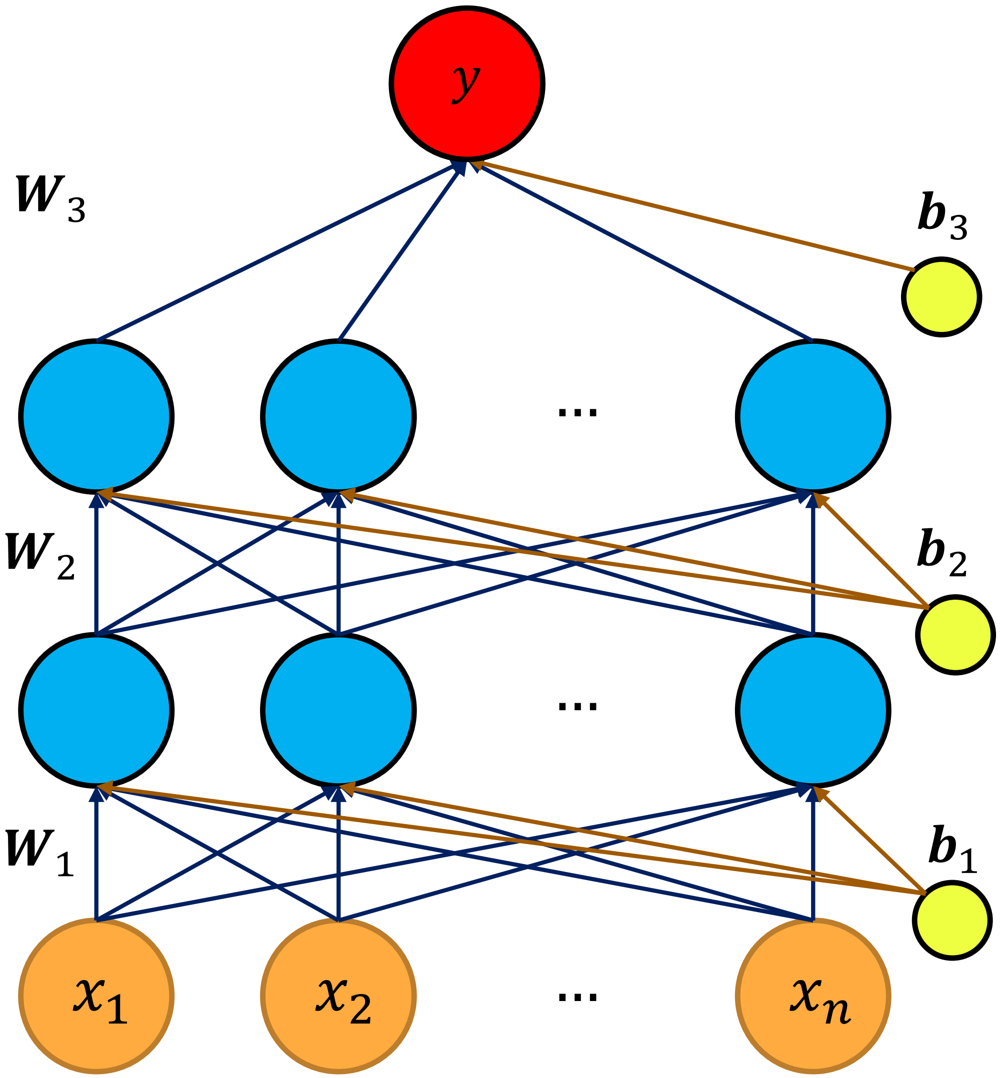
$$\boldsymbol{x} = [x_1 \quad x_2 \quad \dots \quad x_n]^T$$

- $L - 1$ hidden layers with m neurons each
- Output layer (L^{th}) with k neurons
- $\boldsymbol{W}_i$ is the matrix containing the weights between layers $i - 1$ and i $(0 < i \leq L)$
- $\boldsymbol{b}_i$ is the vector representing the biases

$$\boldsymbol{W}_1 = \begin{bmatrix} w_{11}^1 & \dots & w_{1n}^1 \\ \vdots & \ddots & \vdots \\ w_{m1}^1 & \dots & w_{mn}^1 \end{bmatrix}_{m \times n} \qquad \boldsymbol{b}_1 = \begin{bmatrix} b_1^1 \\ \vdots \\ b_m^1 \end{bmatrix}$$

$$\boldsymbol{W}_i = \begin{bmatrix} w_{11}^i & \dots & w_{1m}^i \\ \vdots & \ddots & \vdots \\ w_{m1}^i & \dots & w_{mm}^i \end{bmatrix}_{m \times m} \qquad \boldsymbol{b}_{\text{i}} = \begin{bmatrix} b_1^i \\ \vdots \\ b_m^i \end{bmatrix}$$

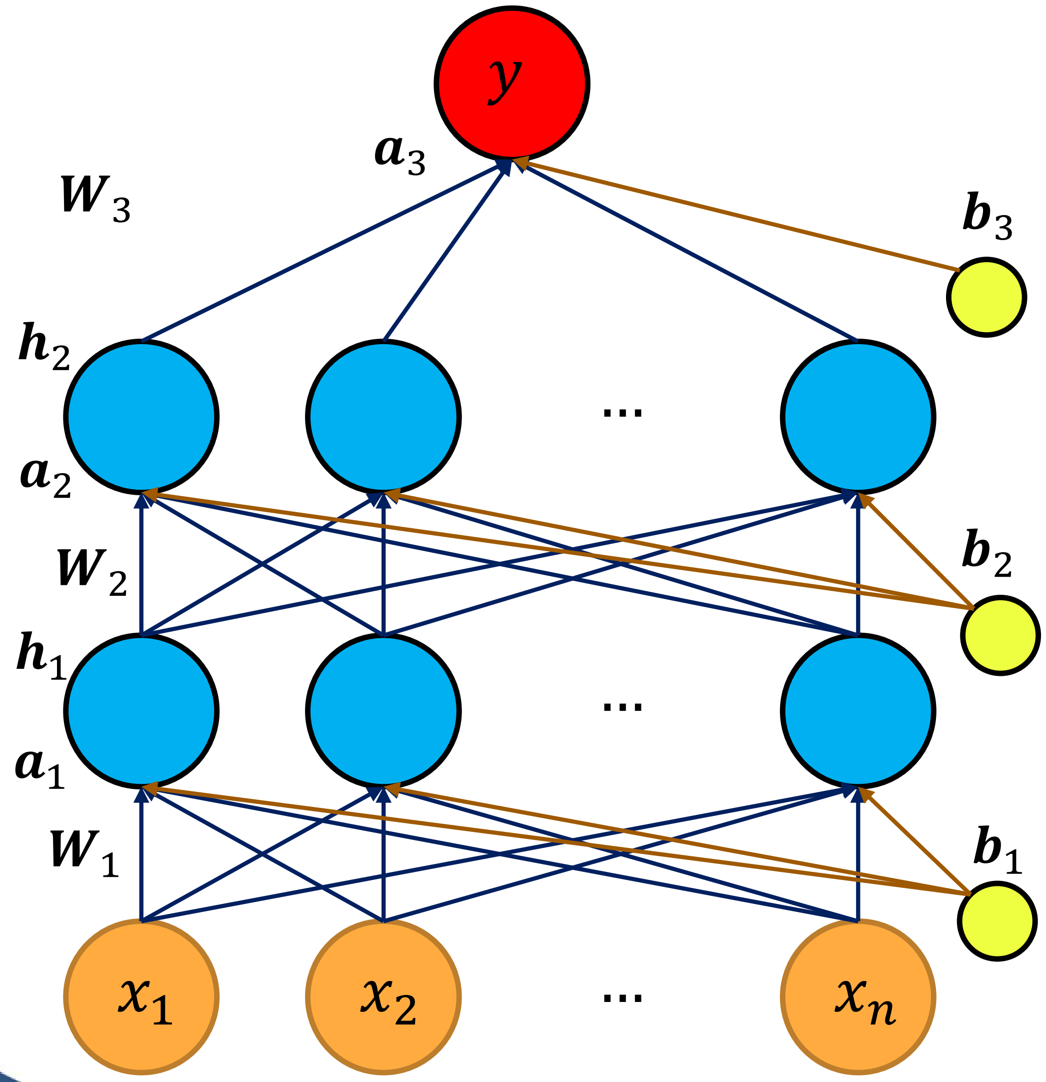
ANN Architecture



$$\boldsymbol{W}_L = \begin{bmatrix} w_{11}^L & \cdots & w_{1m}^L \\ \vdots & \ddots & \vdots \\ w_{k1}^L & \cdots & w_{km}^L \end{bmatrix}_{k\times m} \qquad \boldsymbol{b}_{\mathrm{L}} = \begin{bmatrix} b_1^L \\ \vdots \\ b_k^L \end{bmatrix}$$

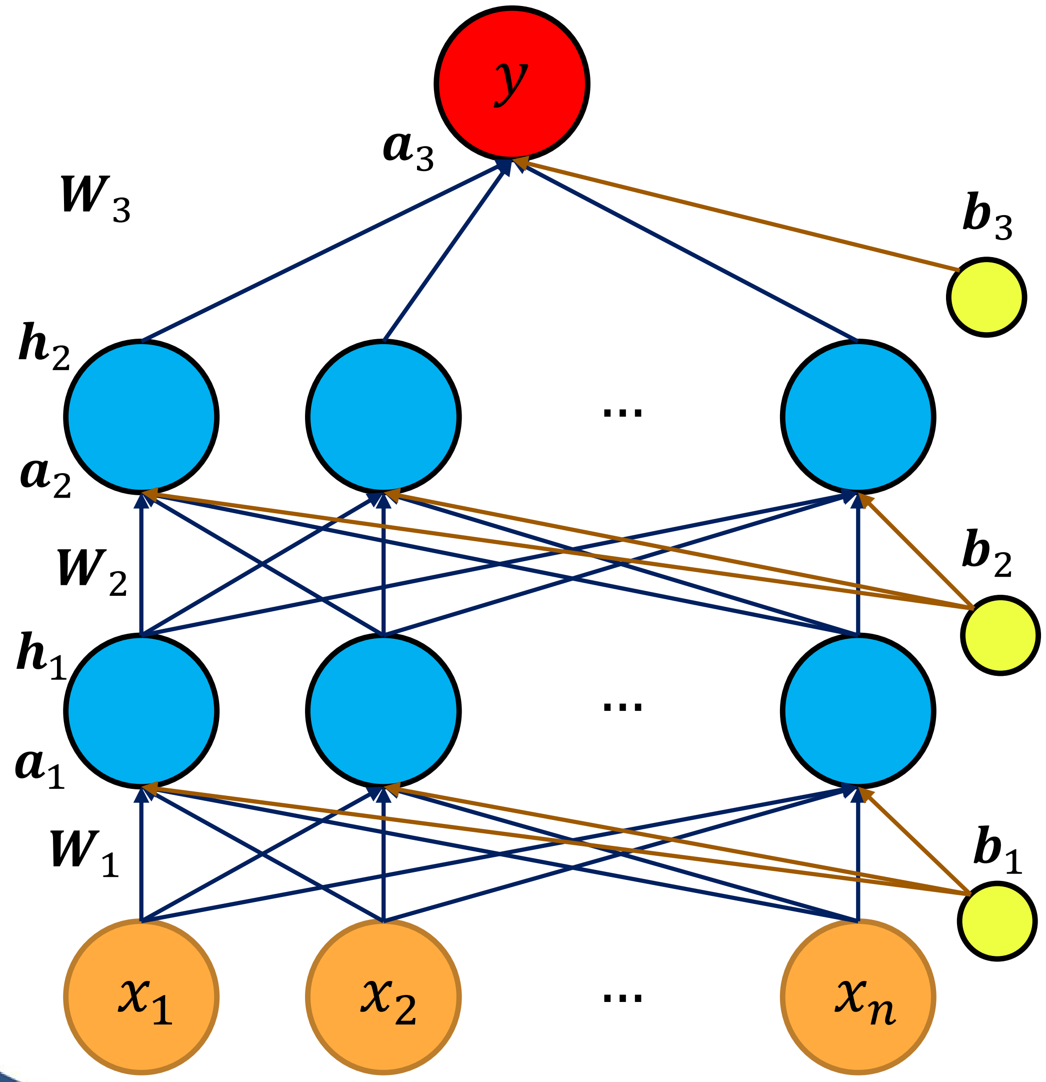
- For a single output, $\boldsymbol{W}_L$ will be a vector and $\boldsymbol{b}_L$ will be a scalar
- Each neuron in hidden and output layers has an activation function
- If there are more number of hidden layers, then ANN is usually referred to as Deep Neural Network (DNN) – Depth refers to the number of layers

DNN Feed Forward Calculation



- Feed forward calculation involves finding the output as a function of input, weights and biases
- Input to activation function at layer 0:
$$\boldsymbol{a}_1 = \boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{b}_1$$
- Activation at hidden layer 1:
$$\boldsymbol{h}_1 = g_h(\boldsymbol{a}_1)$$
- $\boldsymbol{h}_i$ is output vector at layer i
- g_h is the activation function which maps vector $\boldsymbol{a}_i$ to vector $\boldsymbol{h}_i$
- Input to activation function at hidden layer i:
$$\boldsymbol{a}_i = \boldsymbol{W}_i\boldsymbol{h}_{i-1} + \boldsymbol{b}_i$$
- Activation at hidden layer i:
$$\boldsymbol{h}_i = g_h(\boldsymbol{a}_i) = g_h(\boldsymbol{W}_i\boldsymbol{h}_{i-1} + \boldsymbol{b}_i)$$

DNN Feed Forward Calculation



$$\boldsymbol{a}_1 = \boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1$$
$$\boldsymbol{h}_1 = g_h(\boldsymbol{a}_1)$$
$$\boldsymbol{a}_i = \boldsymbol{W}_i \boldsymbol{h}_{i-1} + \boldsymbol{b}_i$$
$$\boldsymbol{h}_i = g_h(\boldsymbol{a}_i) = g_h(\boldsymbol{W}_i \boldsymbol{h}_{i-1} + \boldsymbol{b}_i)$$

- Input to activation function at output layer L:

$$\boldsymbol{a}_L = \boldsymbol{W}_L \boldsymbol{h}_{L-1} + \boldsymbol{b}_L$$

- Activation at output layer:

$$\hat{\boldsymbol{y}} = g_o(\boldsymbol{a}_L) = g_o(\boldsymbol{W}_L \boldsymbol{h}_{L-1} + \boldsymbol{b}_L)$$

- Model (function) being approximated by the DNN (assuming L=3):

$$\hat{\boldsymbol{y}} = g_o(\boldsymbol{W}_3(\boldsymbol{W}_2(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2) + \boldsymbol{b}_3)$$
$$\hat{\boldsymbol{y}} = f(\boldsymbol{x})$$

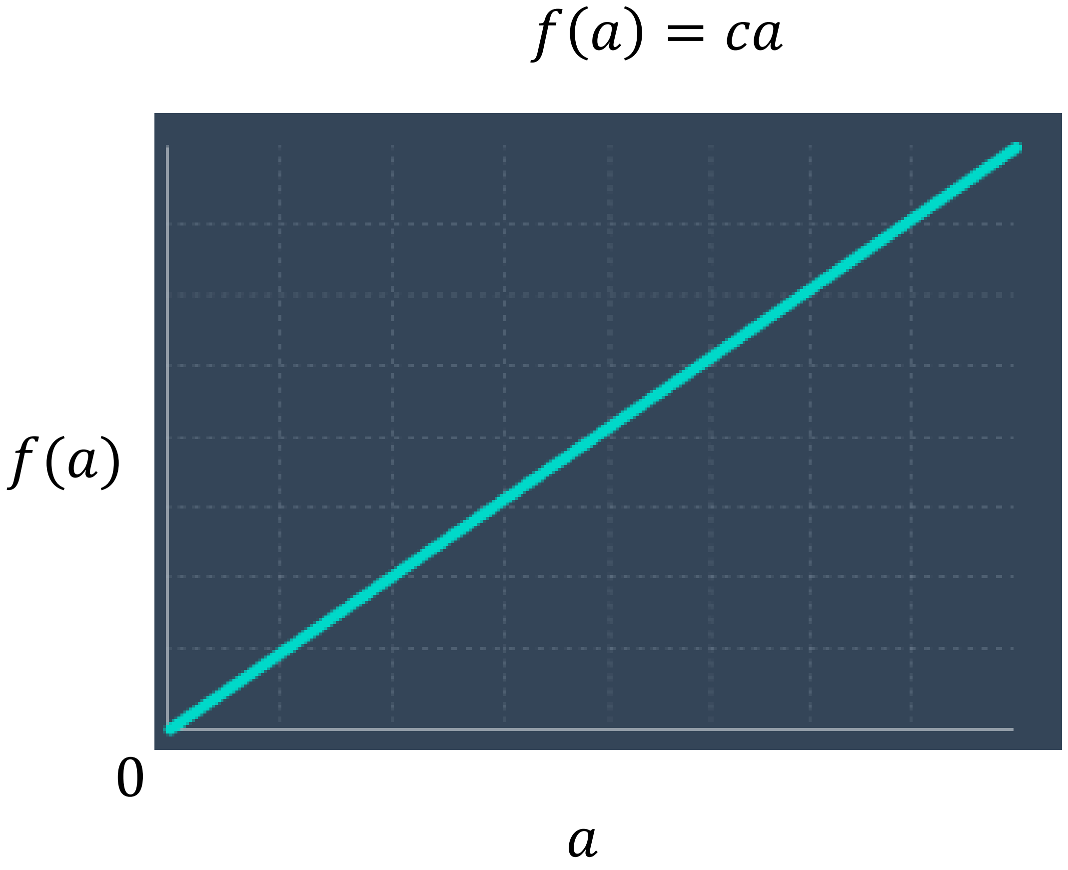
Types of Activation Functions

Activation Function

- Activation function is like a gate between the input and output of a neuron
- Purpose: To introduce non-linearity into the model and enable learning complex functions (models)
- It affects the DNN output, accuracy and convergence
- Types of activation function:
  1. Linear activation function
  2. Sigmoid activation function
  3. Tanh activation function
  4. Relu activation function
  5. Softmax activation function

Linear Activation Function

- Output is directly proportional to the input
$$f(a) = ca$$
- Output can take any real number
- Gradient is always constant and does not depend on the input
- Generally used in the output layer of regression problem



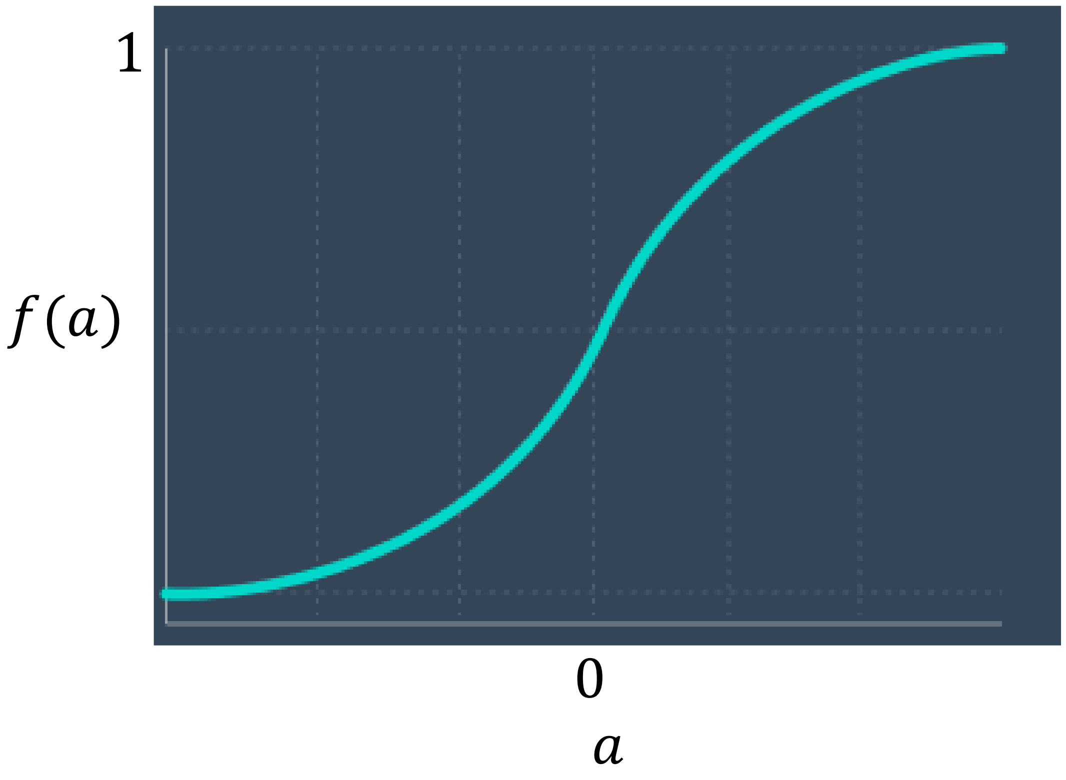
Sigmoid Activation Function

- Any value of input is mapped to a value between 0 and 1

$$f(a) = \frac{1}{1 + e^{-a}}$$

- Gradient is close to zero when the output is close to 0 or 1
- Useful when the expected output is a probabilistic value between 0 and 1

$$f(a) = \frac{1}{1 + e^{-a}}$$



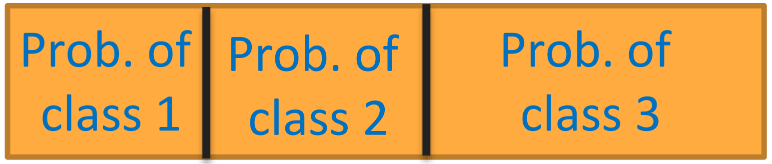
Softmax Activation Function

- Sigmoid function gives a value between 0 and 1, and can be used for binary classification
- However, sigmoid cannot be used to output multiple probability values which add up to 1 (multi-class)
- Softmax function is an extension of sigmoid function
- Softmax calculates the relative probabilities of multiples classes and ensures that total probability is 1

Total Probability = 1

Binary Classification

Total Probability = 1



Multi-class Classification

Input to output layer of a DNN

$$\boldsymbol{a} = [a_1 \quad a_2 \quad \dots \quad a_k]$$

$$f(a_i) = \frac{e^{a_i}}{\sum_{i=1}^{k} e^{a_i}}$$

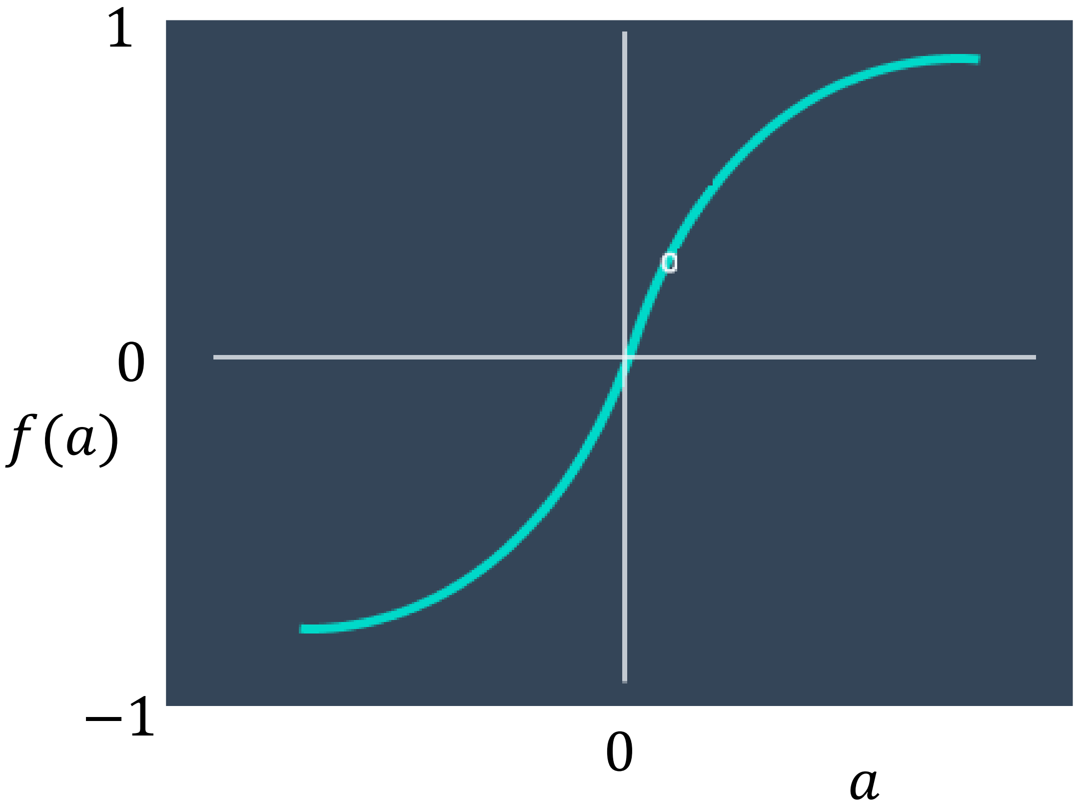
Tanh Activation Function

- Any value of input is mapped to a value between -1 and 1

$$f(a) = \frac{e^{x} - e^{-a}}{e^{a} + e^{-a}}$$

- Positive values between 0 and 1
- Negative values between -1 and 0
- Gradient is close to zero when the output is close to -1 or 1

$$f(a) = \frac{e^{x} - e^{-a}}{e^{a} + e^{-a}}$$



Relu Activation Function

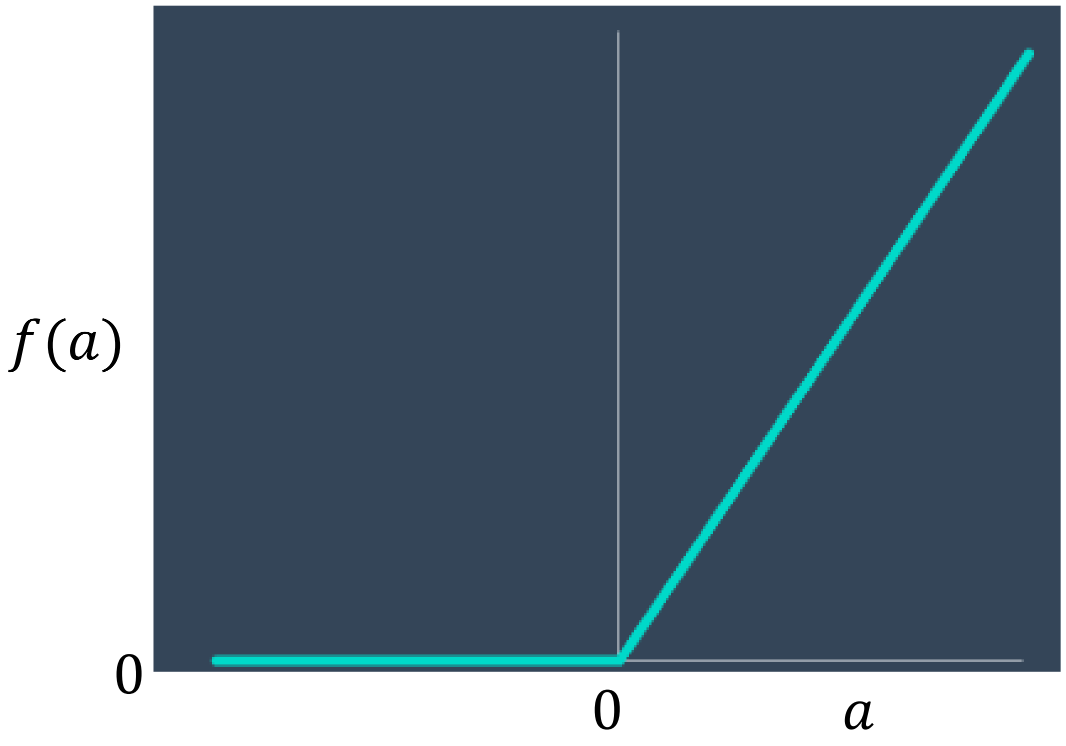
Rectified Linear Unit

- All positive values go through directly while all negative values are mapped to zero

$$f(a) = \max(0, a)$$

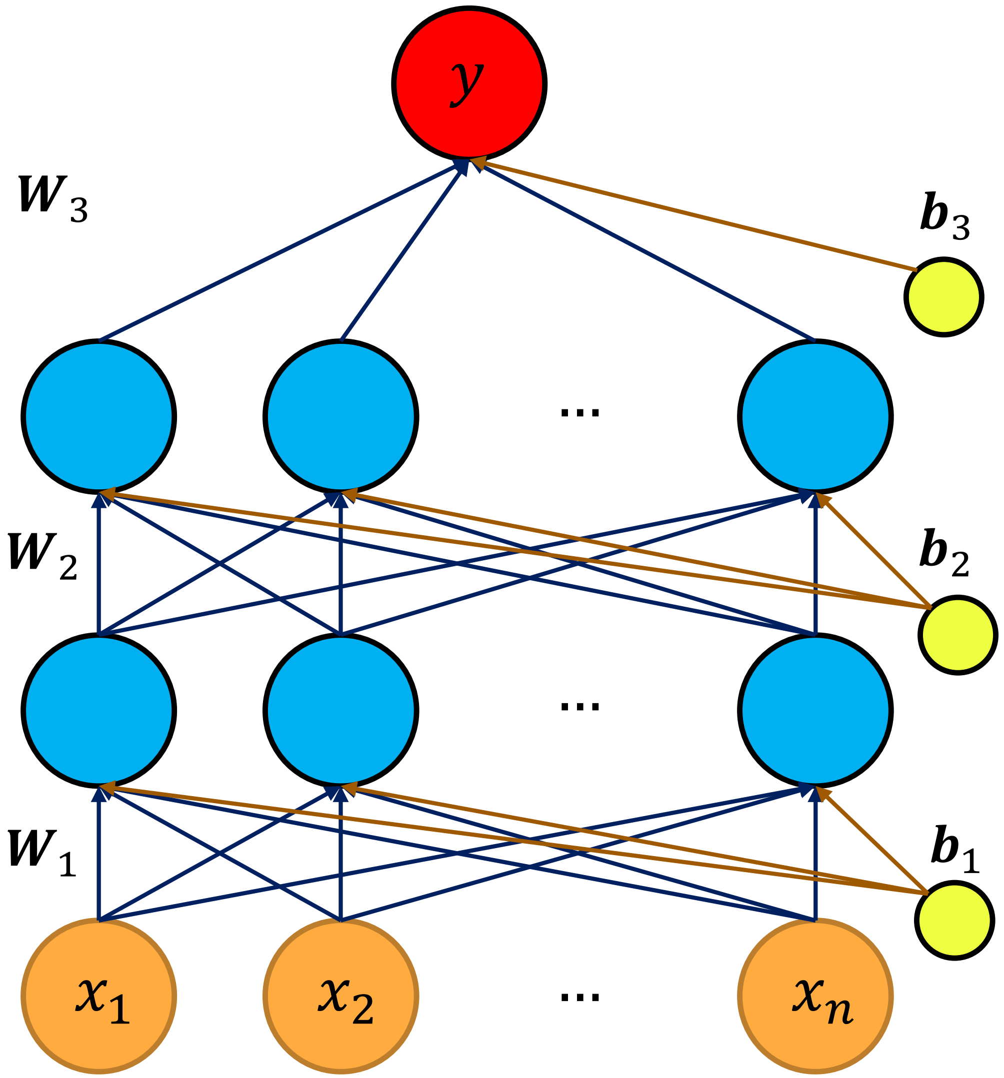
- Gradient is zero when the output $\leq$ 0 and 1 for all positive outputs
- Relu is one of the most popular activation functions and has many variants

$f(a) = \max(0, a)$



Universal Approximation Theorem

Universal Approximation Theorem (UAT)



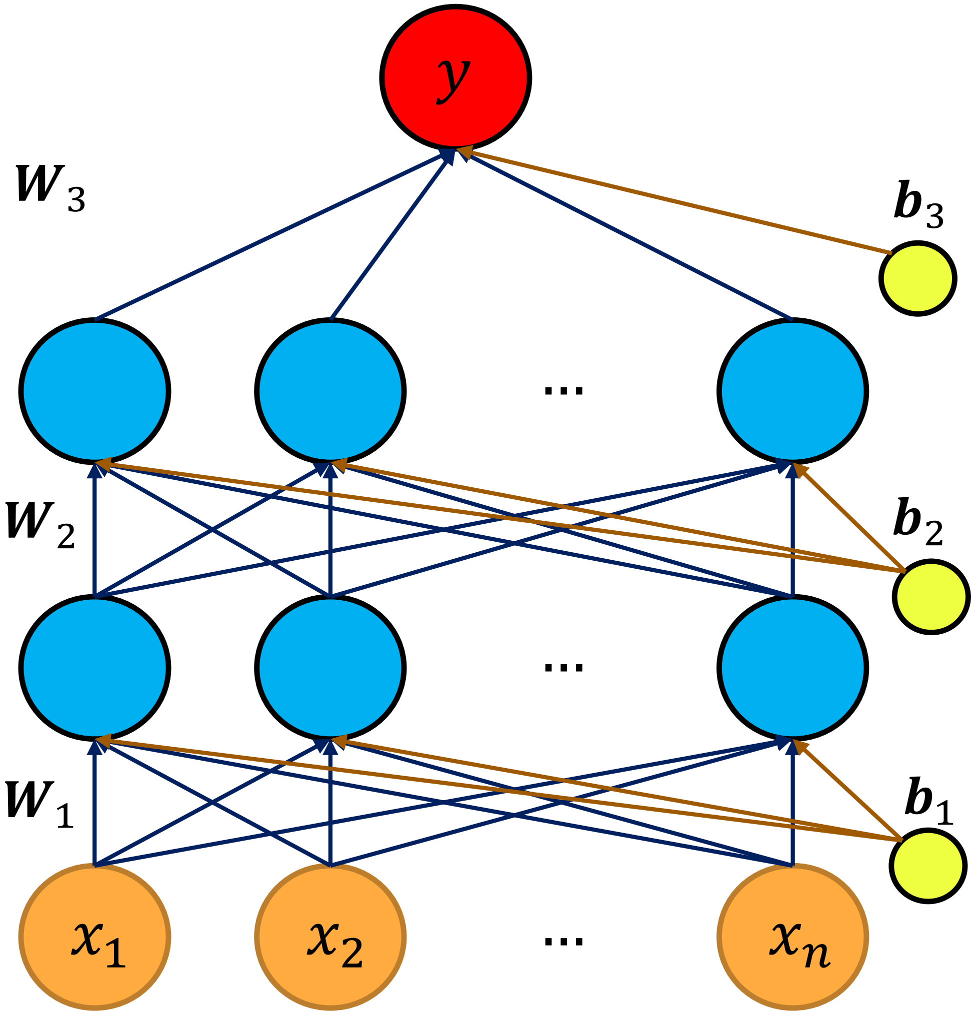
$$y = f(\boldsymbol{x})$$

- UAT establishes that neural networks have a kind of universality in approximating functions
- For any given function of inputs $f(\boldsymbol{x})$, there exists a neural network which can approximate the output
- Holds even when the function has multiple inputs and outputs
- Condition: Activations functions should be non-linear

Ref: Article by Micheal Nelson - [Neural networks and deep learning]

Supervised Learning using DNN

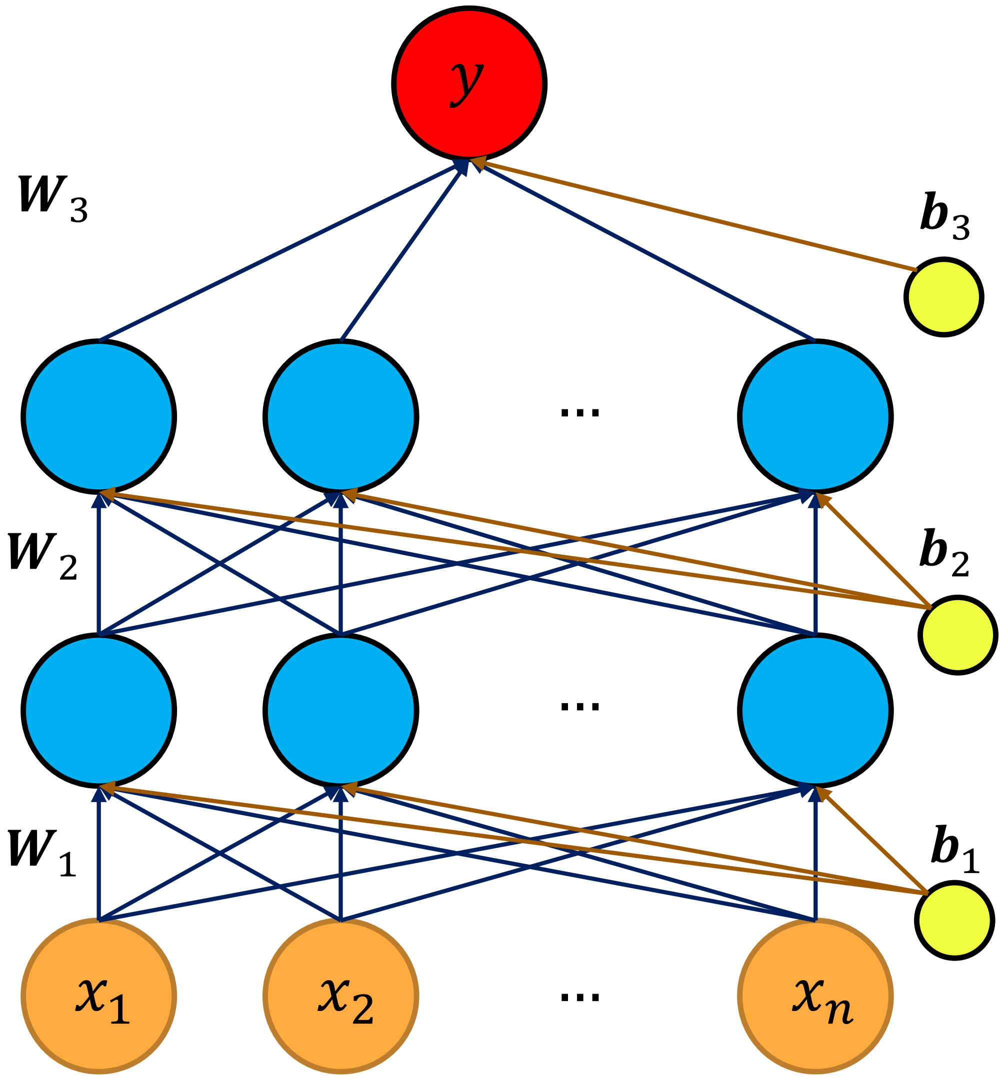
Supervised Learning using DNN



Data for Supervised Learning:

- Inputs: Values of input features – $\boldsymbol{x}$
- Outputs: Values of predicted variables – $\boldsymbol{y}$
  - Regression – Real Numbers
  - Classification – Discrete class or
    - Probability of each class
- DNN is expected to take an input and predict the desired output
- Implies: DNN should approximate a function $f(\boldsymbol{x})$ which maps inputs to outputs

Supervised Learning using DNN



- **Question:** What is a suitable $f(\boldsymbol{x})$ for the given data or task ?
- ✓ **Answer:** Generally, not known and can be a complex function
- **Question:** Can we find the weights and biases which will approximate desired $f(\boldsymbol{x})$?
- ✓ **Answer:** Yes! They can be learnt from data
- **Training a DNN:** Learning the parameters of the DNN (weights and biases) using the given data

Summary

- Deep Learning is a sub-field of machine learning with many applications in diverse areas.
- Functioning of a biological neuron was mathematically modelled to replicate its decision making capability.
- An artificial neural network was developed inspired from the structure of a brain neural network.
- ANN consists of multiple layers of inter-connected neurons which process inputs to give out outputs.
- Universal Approximation theorem establishes that there always exists an ANN which can approximate any function of any complexity.
- An ANN can be trained to map inputs to desired outputs by learning the weights and biases.

THANK YOU