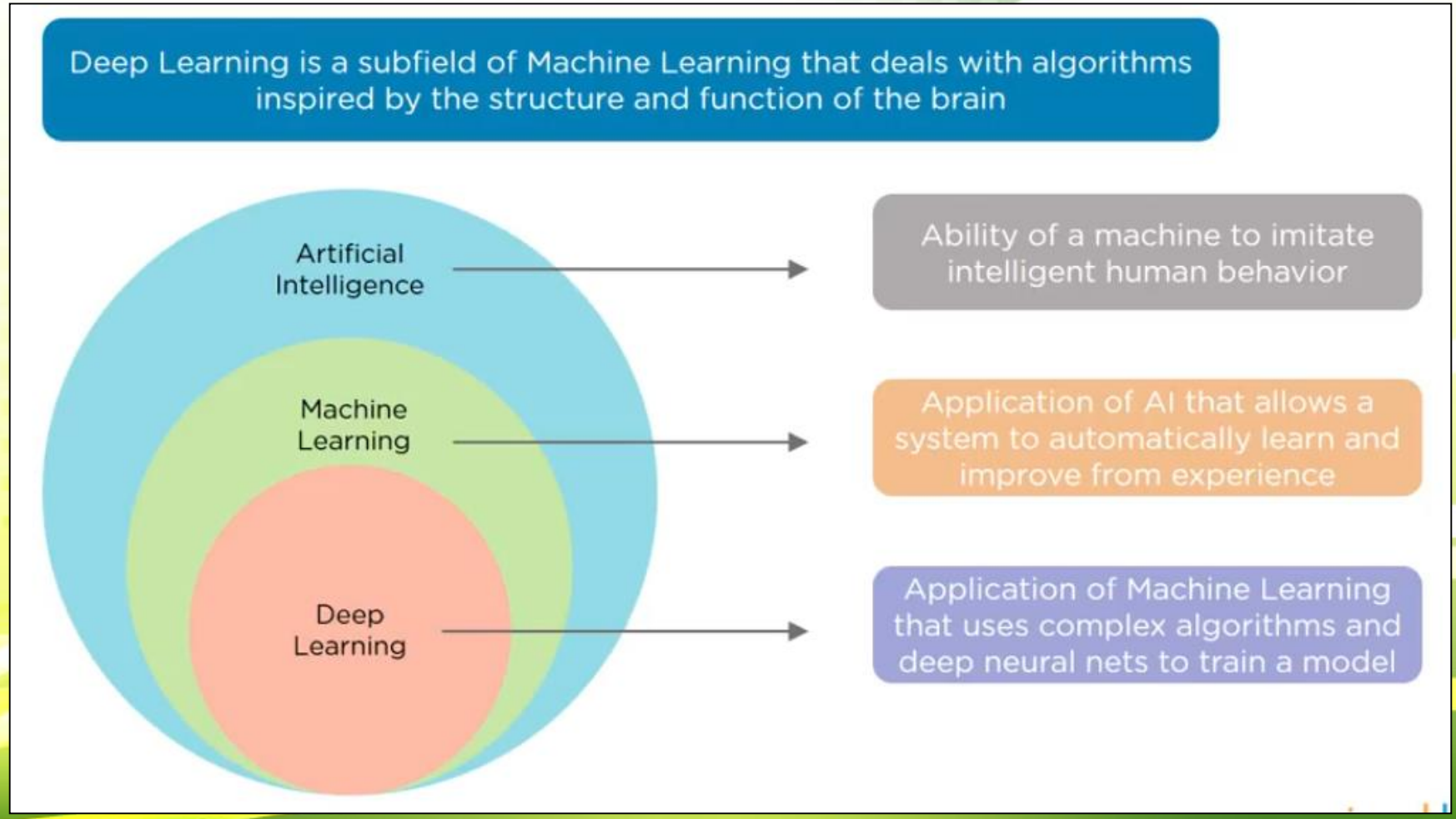
# NEURAL NETWORK

3rd Sem, MCA

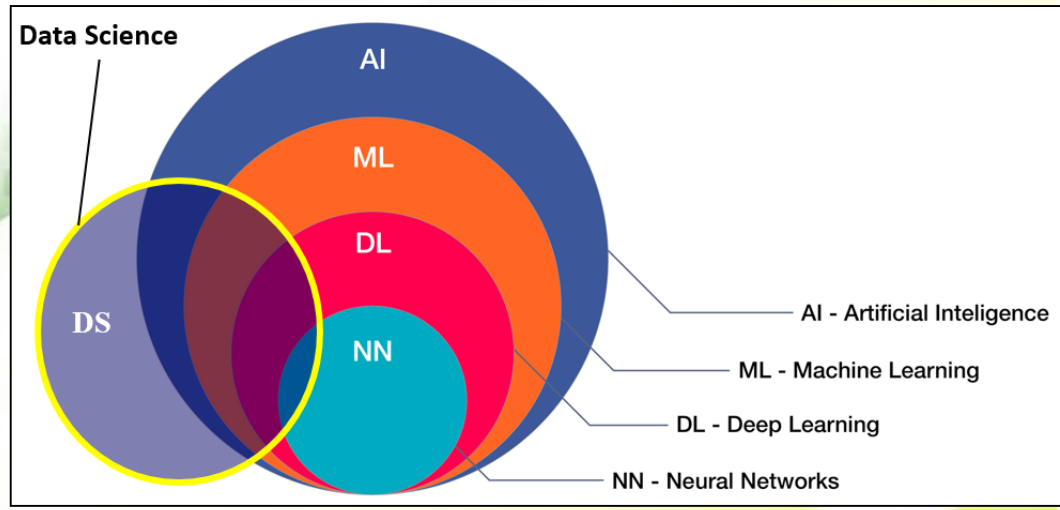
# Contents

- What is Deep Learning
- Why DL
- DL applications
- What is ANN Neural Network
- How ANN works
- Activation function
- Working of Neural Network

# Deep Learning



# Neural Network

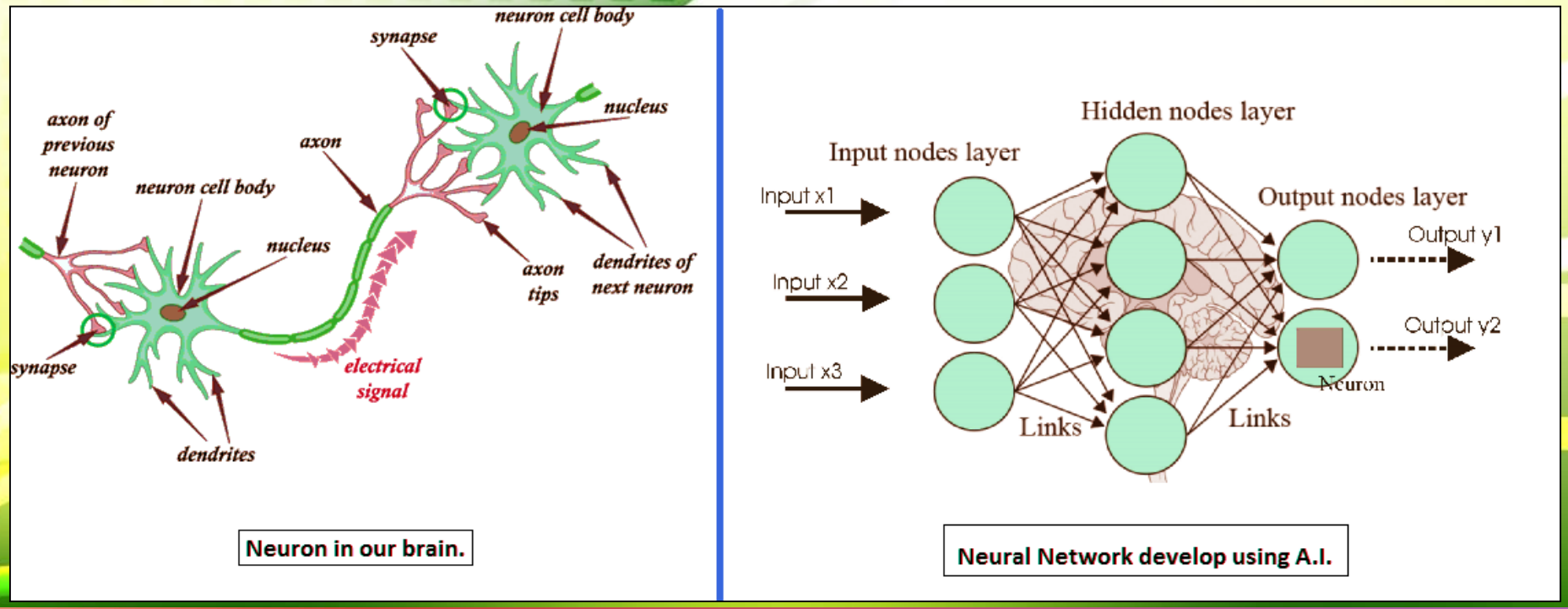



- NN enables DL.
- Since 1890's
- ML/DL uses the NN algorithm to do their work.
- NN is specific group of algorithms used for ML that models the data using graphs of Artificial Neurons.
- Neurons are mathematical models that "*mimics approximately how a neuron in human brain works*".
- Series of algorithms that help to recognize underlying relationships in a set of data through a process that mimics the way human brain operates.

# Neural Network

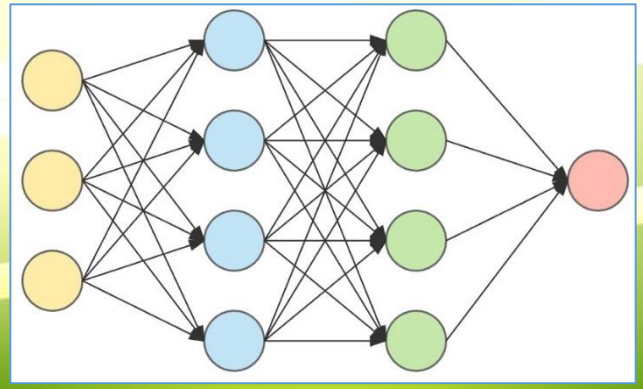
- Input Layer
- Hidden layer
- Output layer

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates.

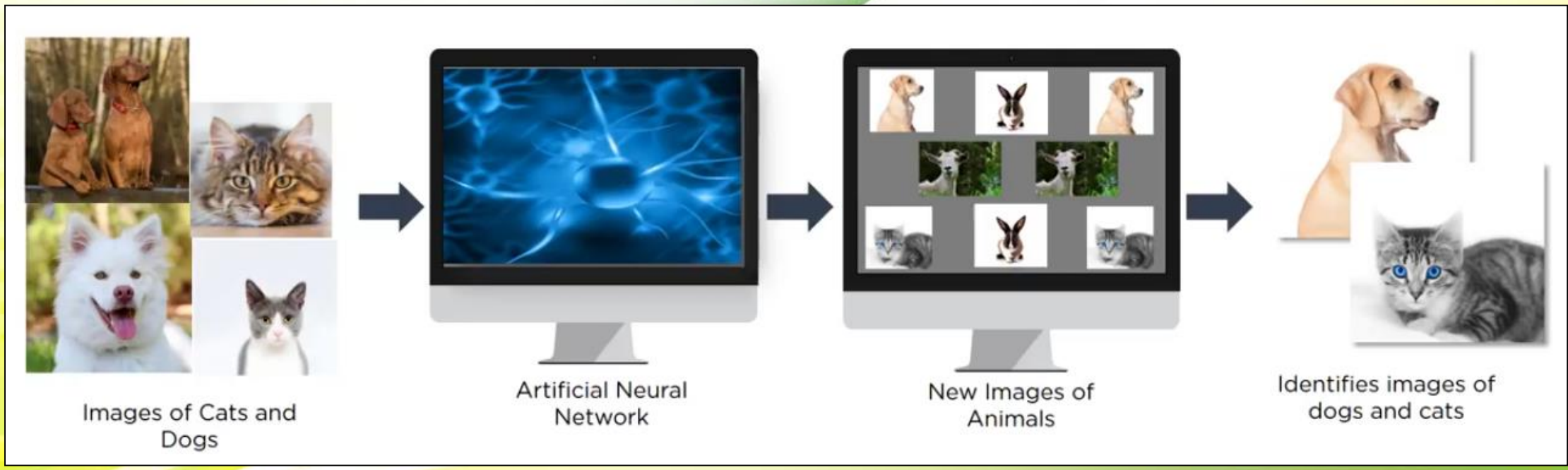


# Neural Network

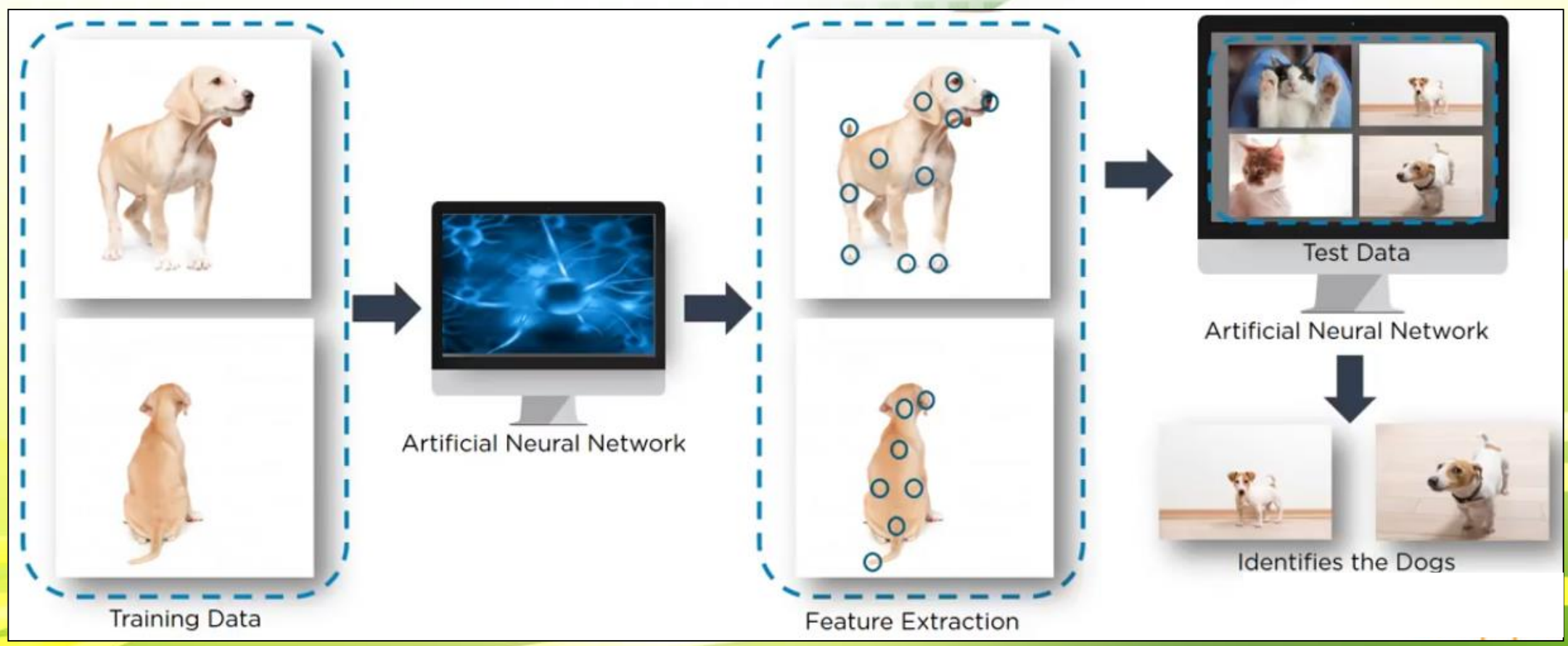
- "Neuron" in neural network is a mathematical function that collects and classifies information according to a specific architecture.
- NN is a strong statistical tool for classification and regression analysis.
- NN contains layers of interconnected nodes.
- Each node is a perceptron and is similar to a multiple linear regression.
- Perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear.



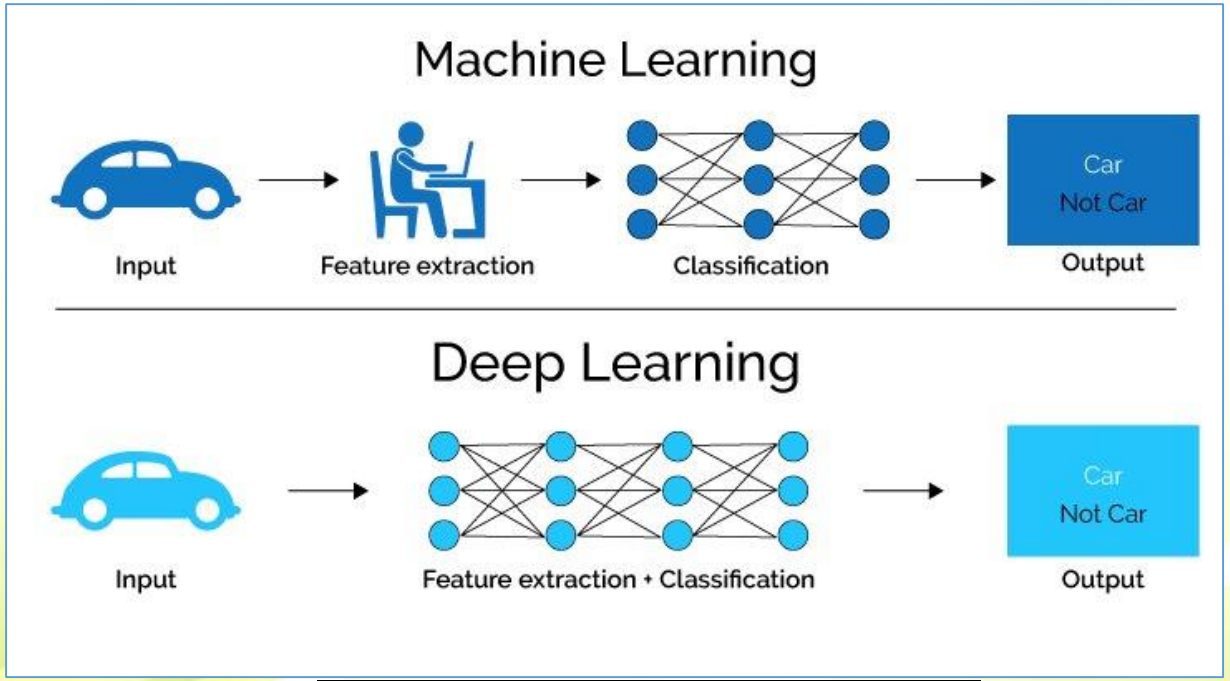
# Deep Learning Application ___ Image Recognition

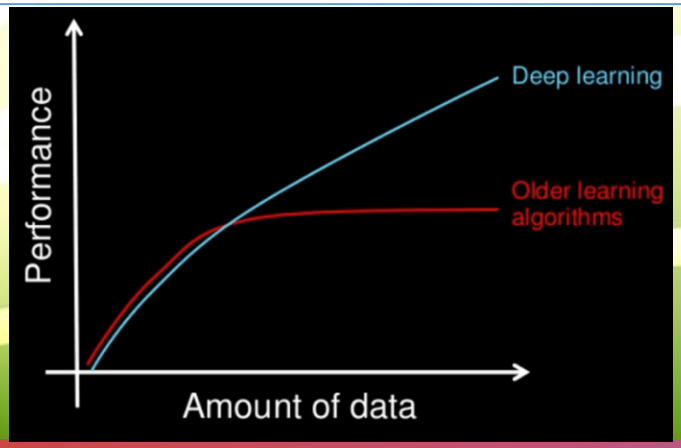


# Deep Learning Application ___ Image Recognition

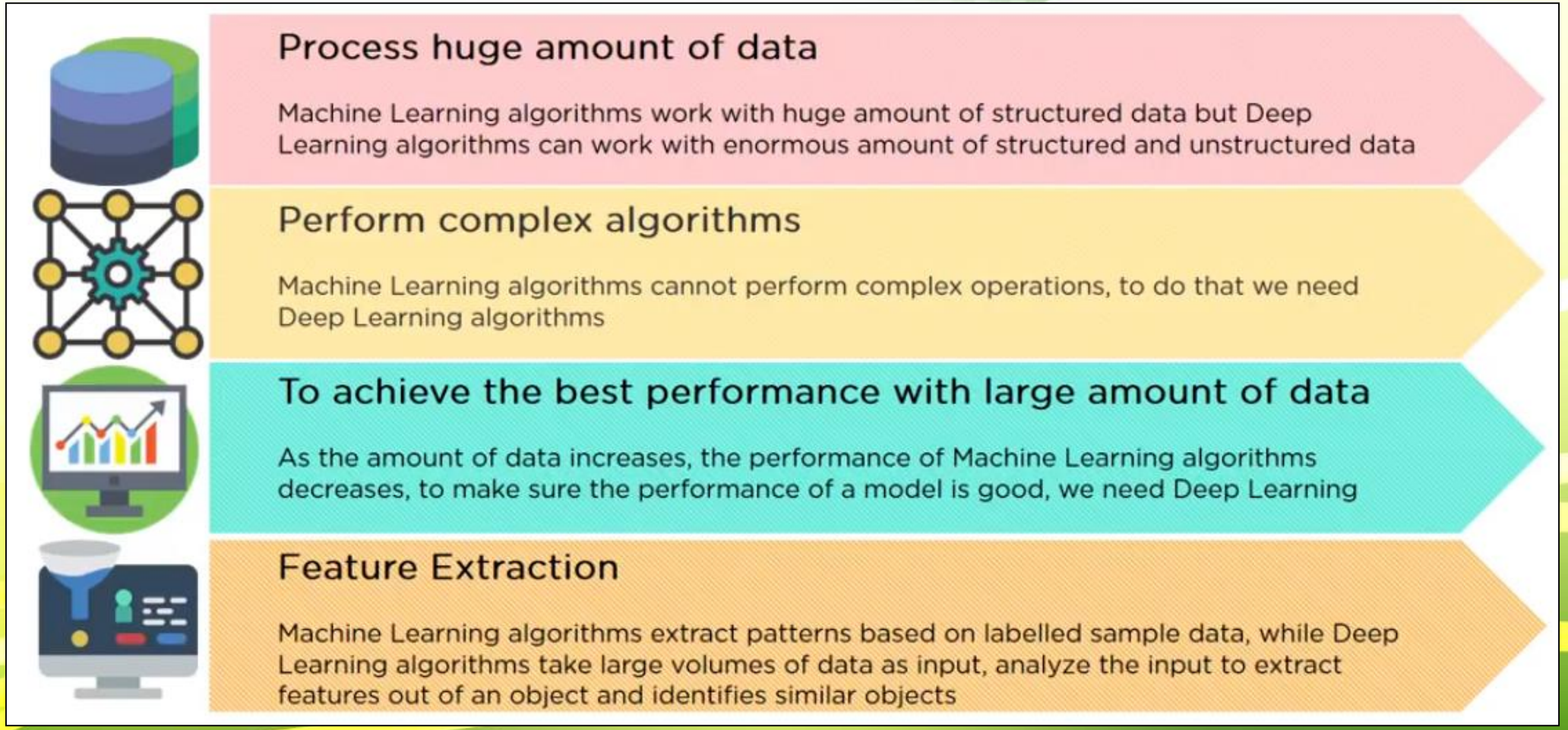


# Deep Learning __ Advantages



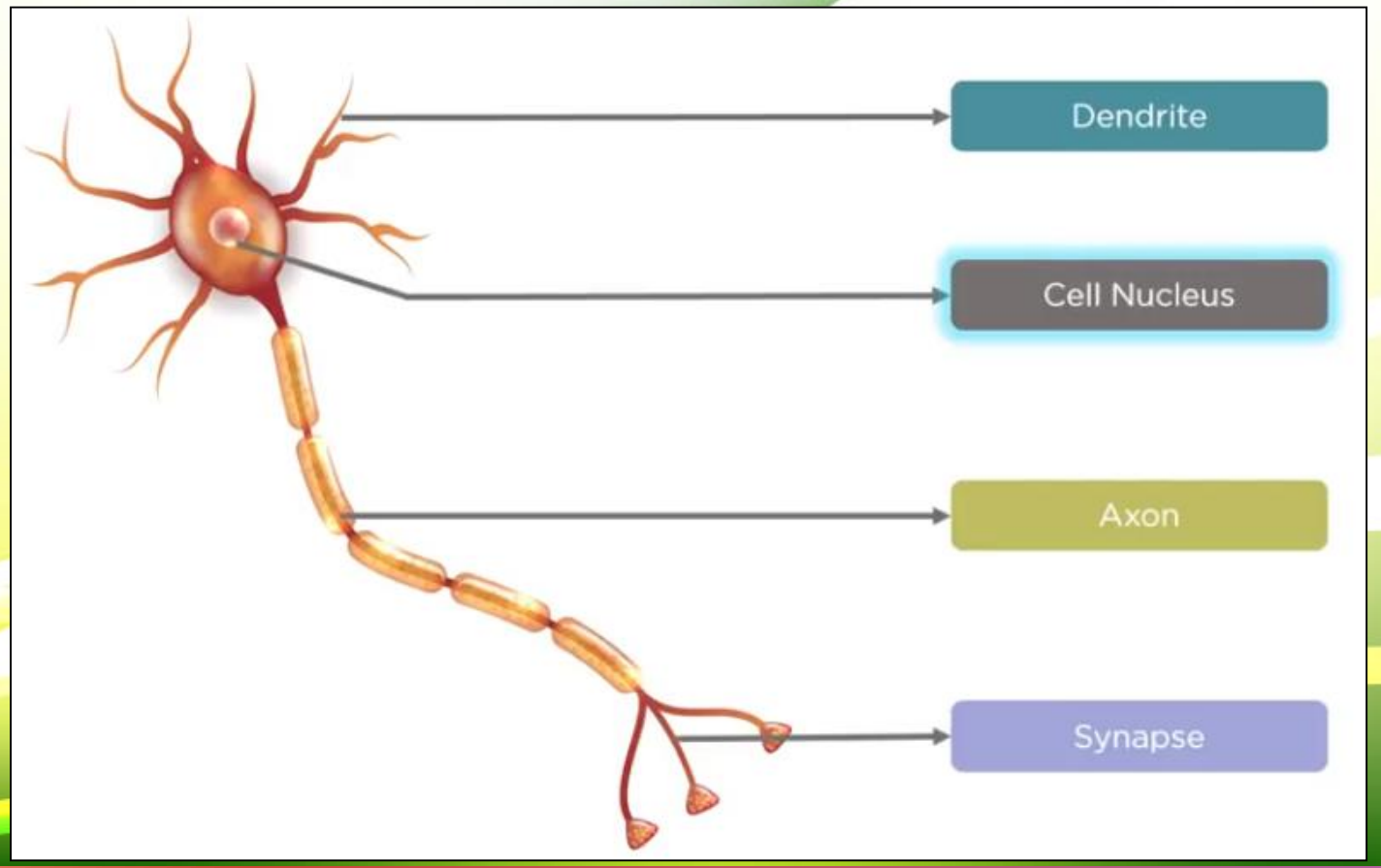



# Deep Learning __ Advantages

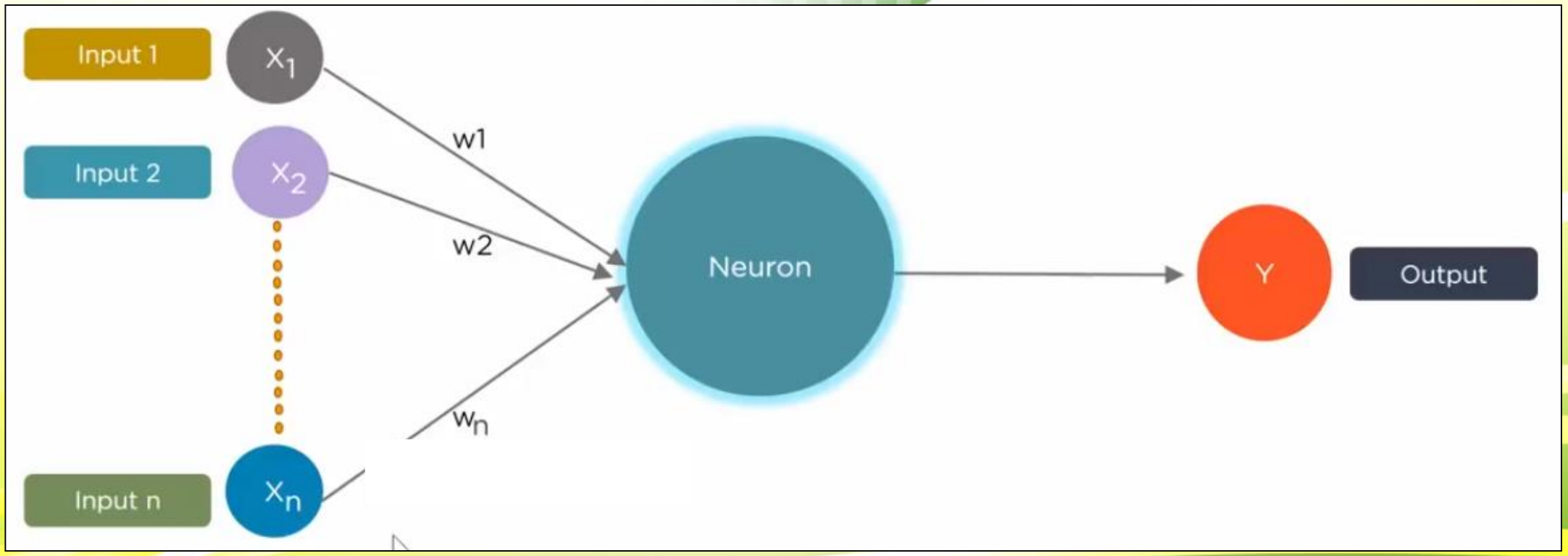


# Deep Learning __ Neural Network

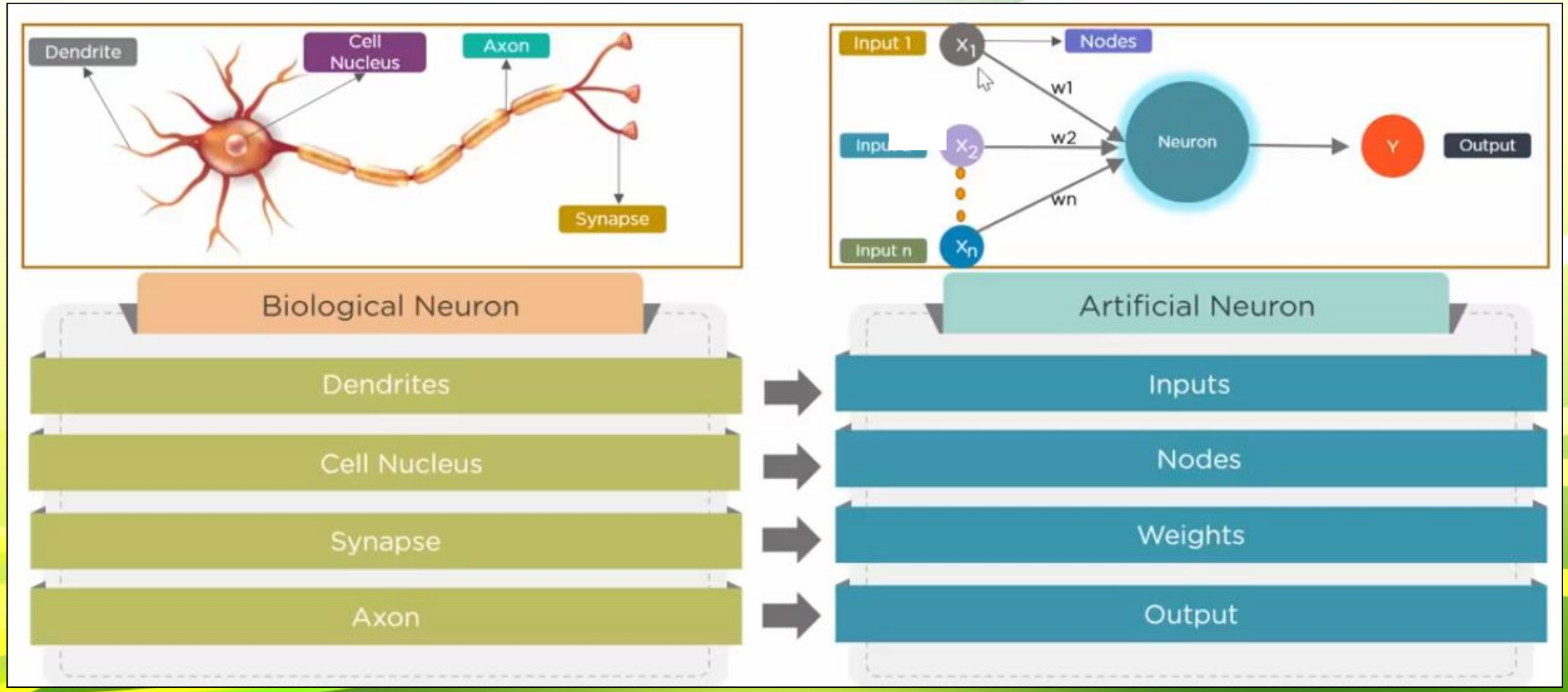
- DL is based on functioning of a Human brain.
- Biological Neural Network is as follows;



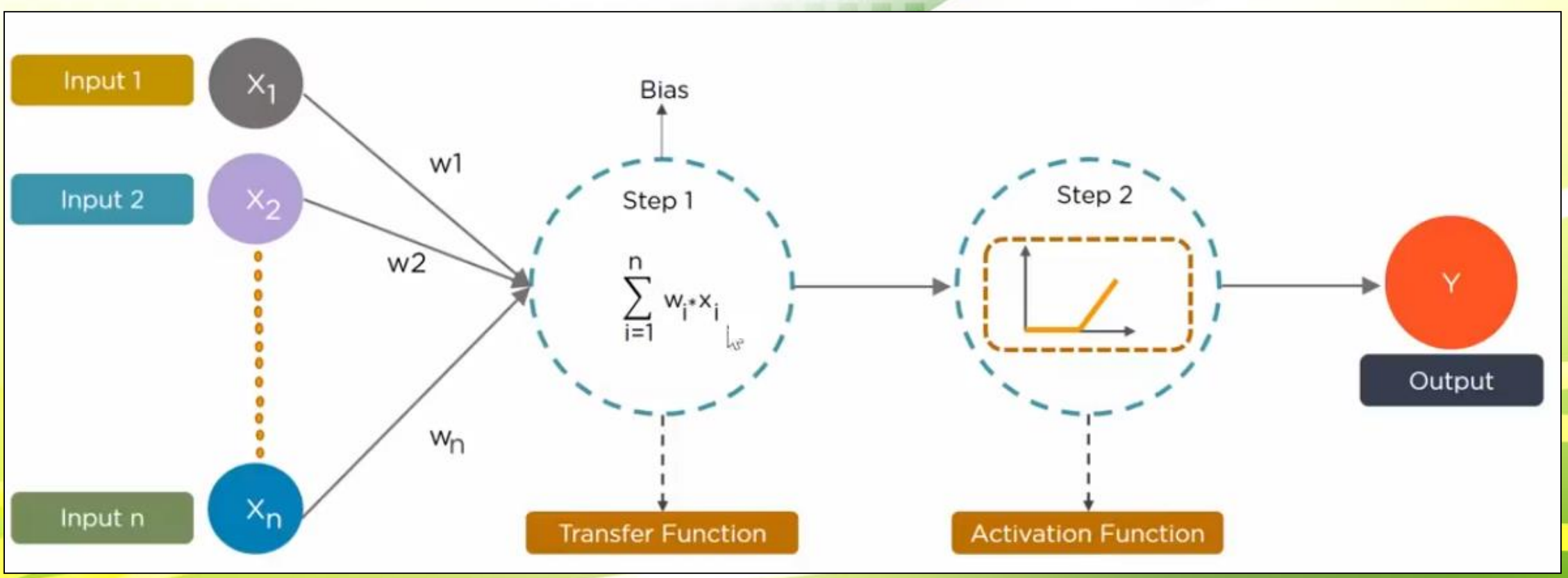
# Deep Learning___ Artificial Neural Network



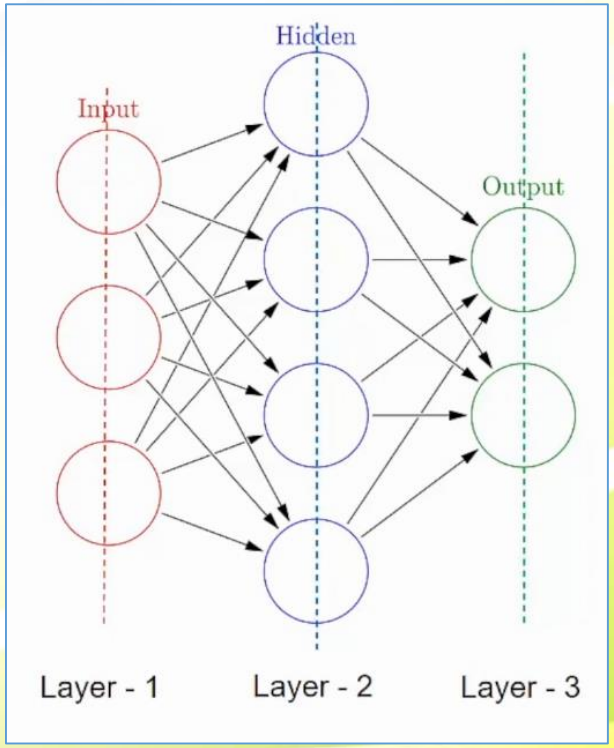
# Deep Learning __ NN vs ANN



# Deep Learning __ ANN



# Deep Learning __ ANN

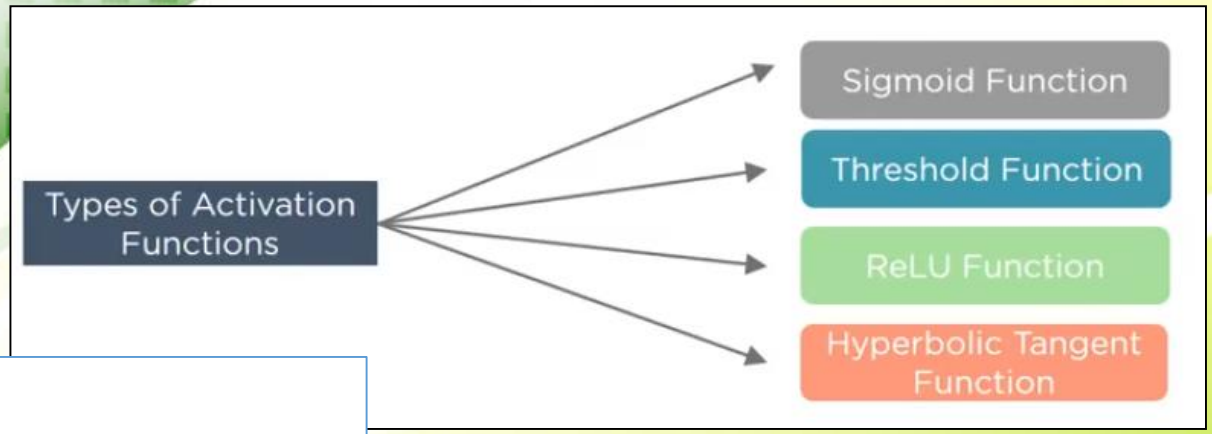


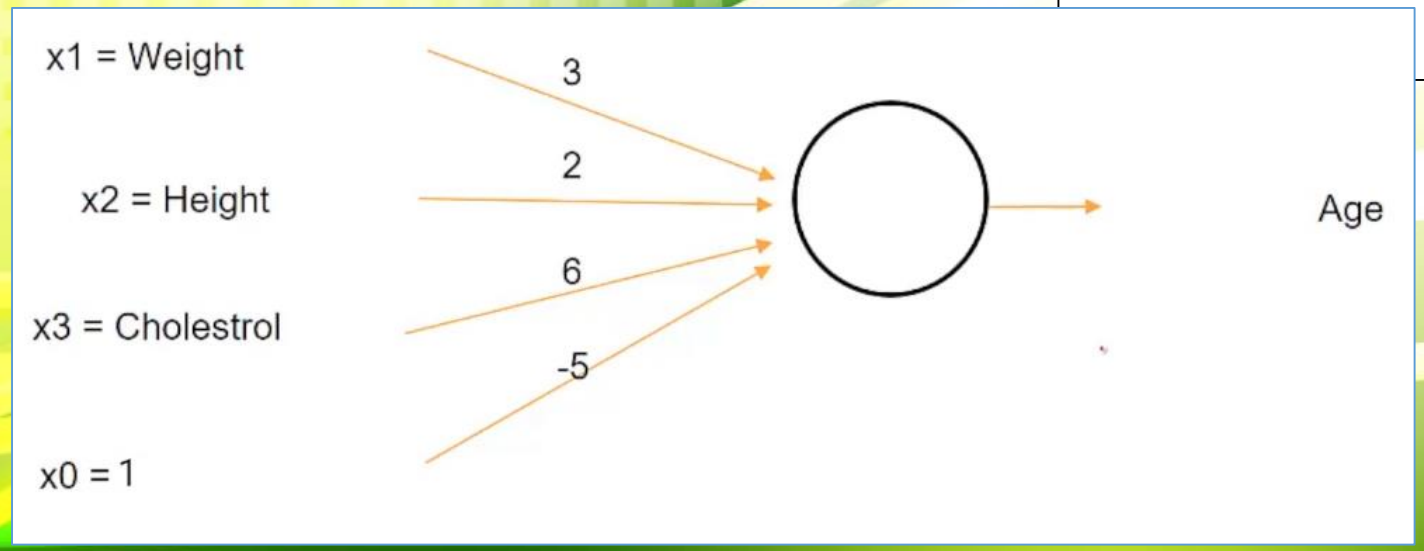

- NN is made up of vertically stacked components called Layers.
- 3 types of layers in a NN; Input, Hidden, Output.

- **Input Layer:** First layer. Accept data and pass it to rest of the network.
- **Hidden Layer:** Second level of layer. One or more in number for a NN.
  - Responsible for excellent performance and complexity of neural networks.
  - Perform multiple functions (data transformation, automatic feature creation, etc.
- **Output layer:** Last type of layer. Holds result or output of the problem.

# Deep Learning __ ANN

- Activation function takes the "weighted sum of inputs and the bias" as the input to the function and decides whether it should be fired or not.

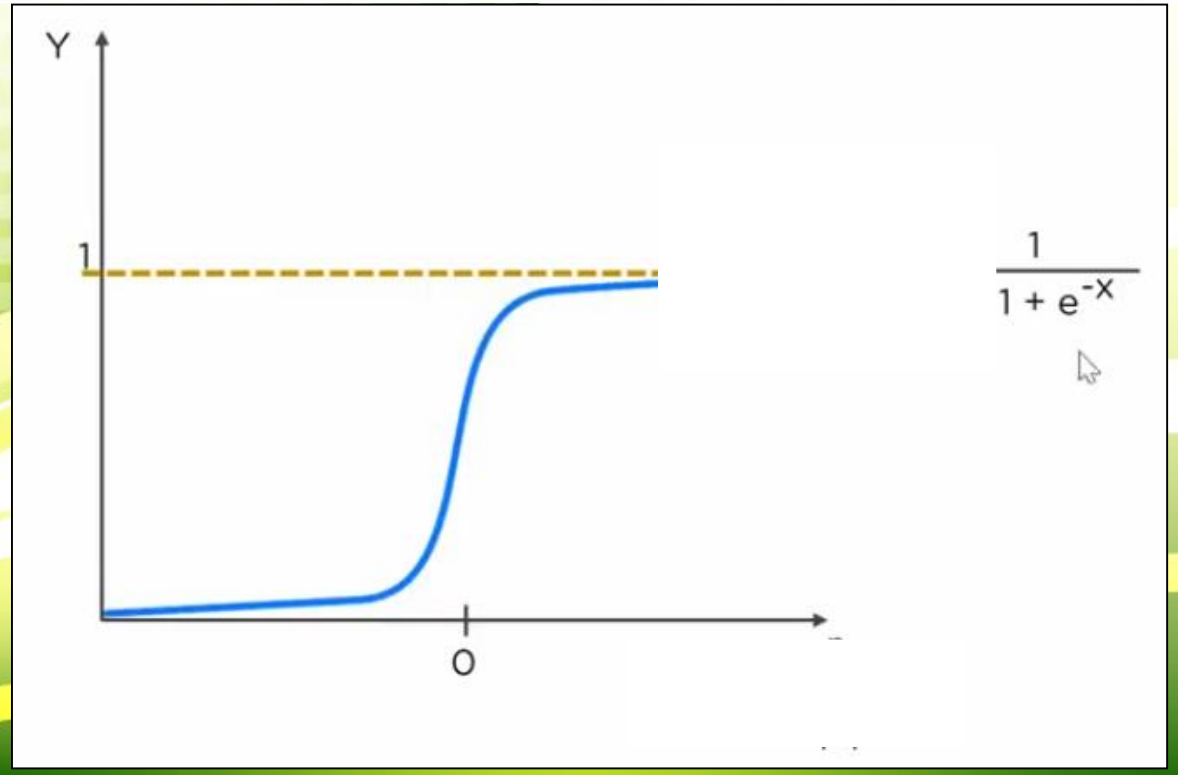


# Deep Learning ___ ANN __ Activation function

Sigmoid function

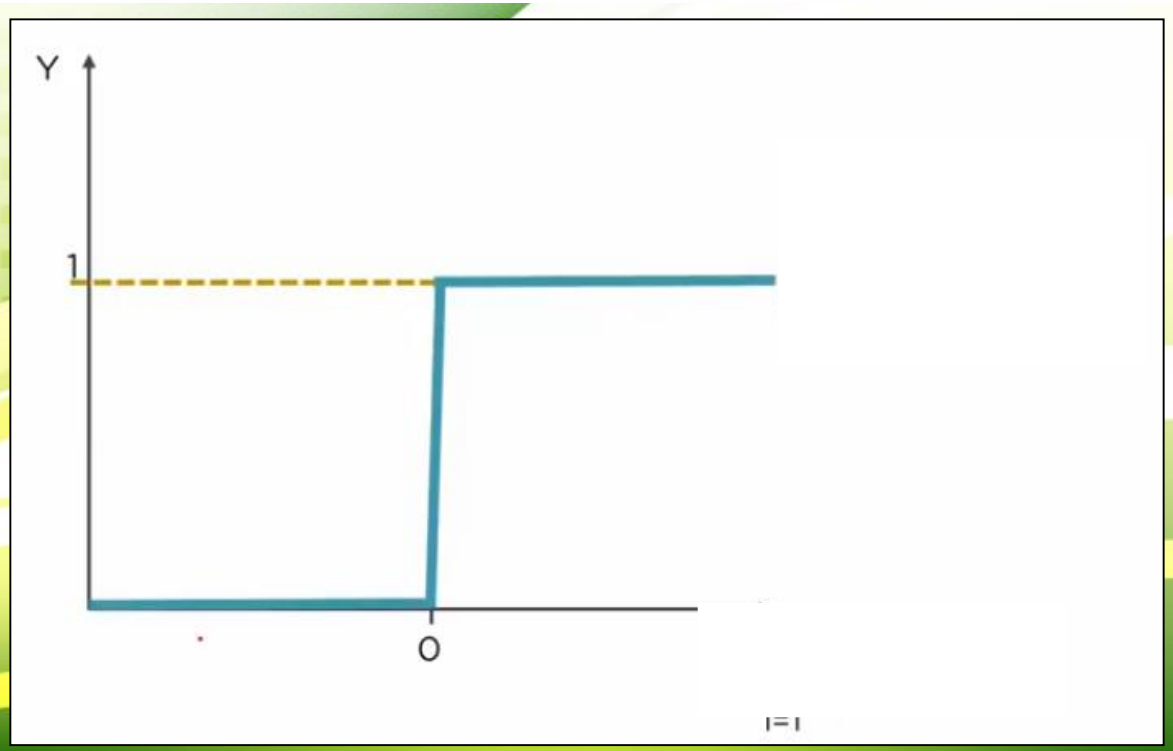
Used for models where we have to predict the probability as an output. It exists between 0 and 1.



# Deep Learning ___ ANN __ Activation function

Threshold function

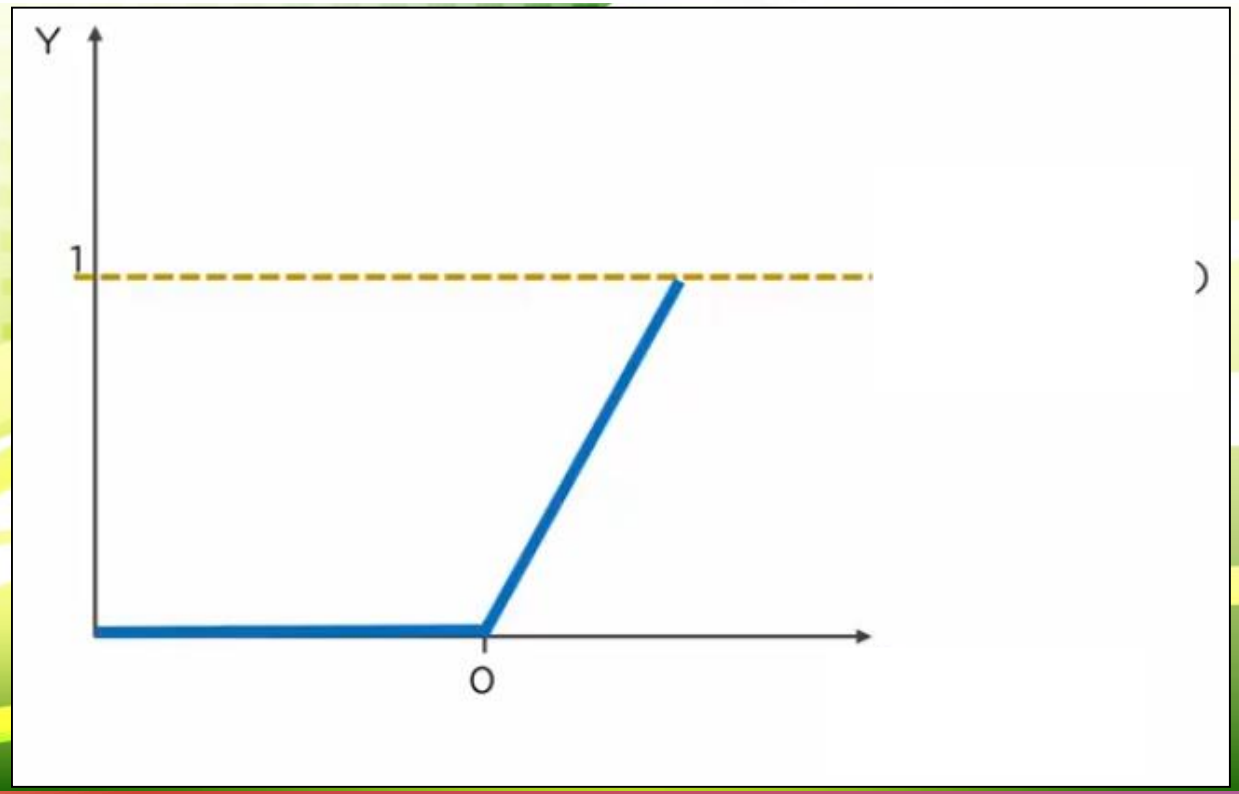
It is a threshold based activation function. If Y value is greater than a certain value, the function is activated and fired else not.



# Deep Learning ___ ANN __ Activation function

ReLU function

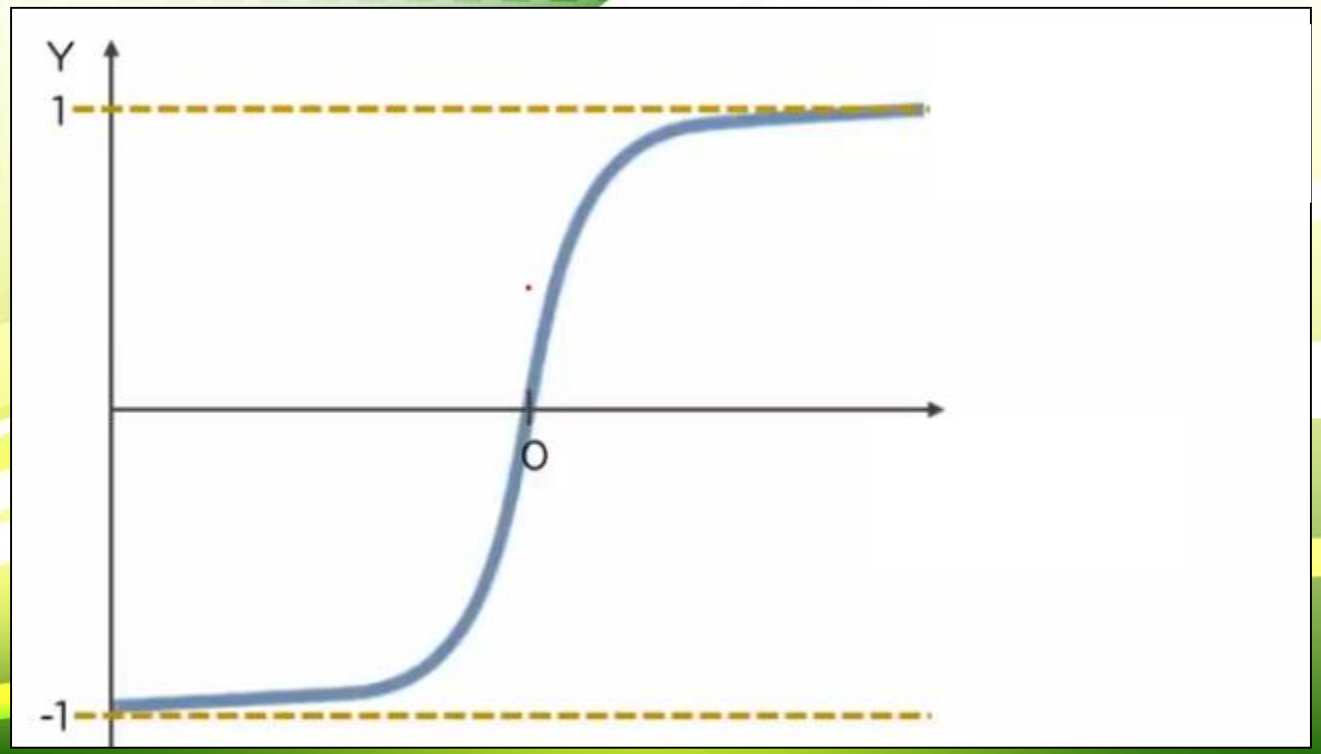
It is the most widely used Activation function and gives an output of X if X is positive and 0 otherwise



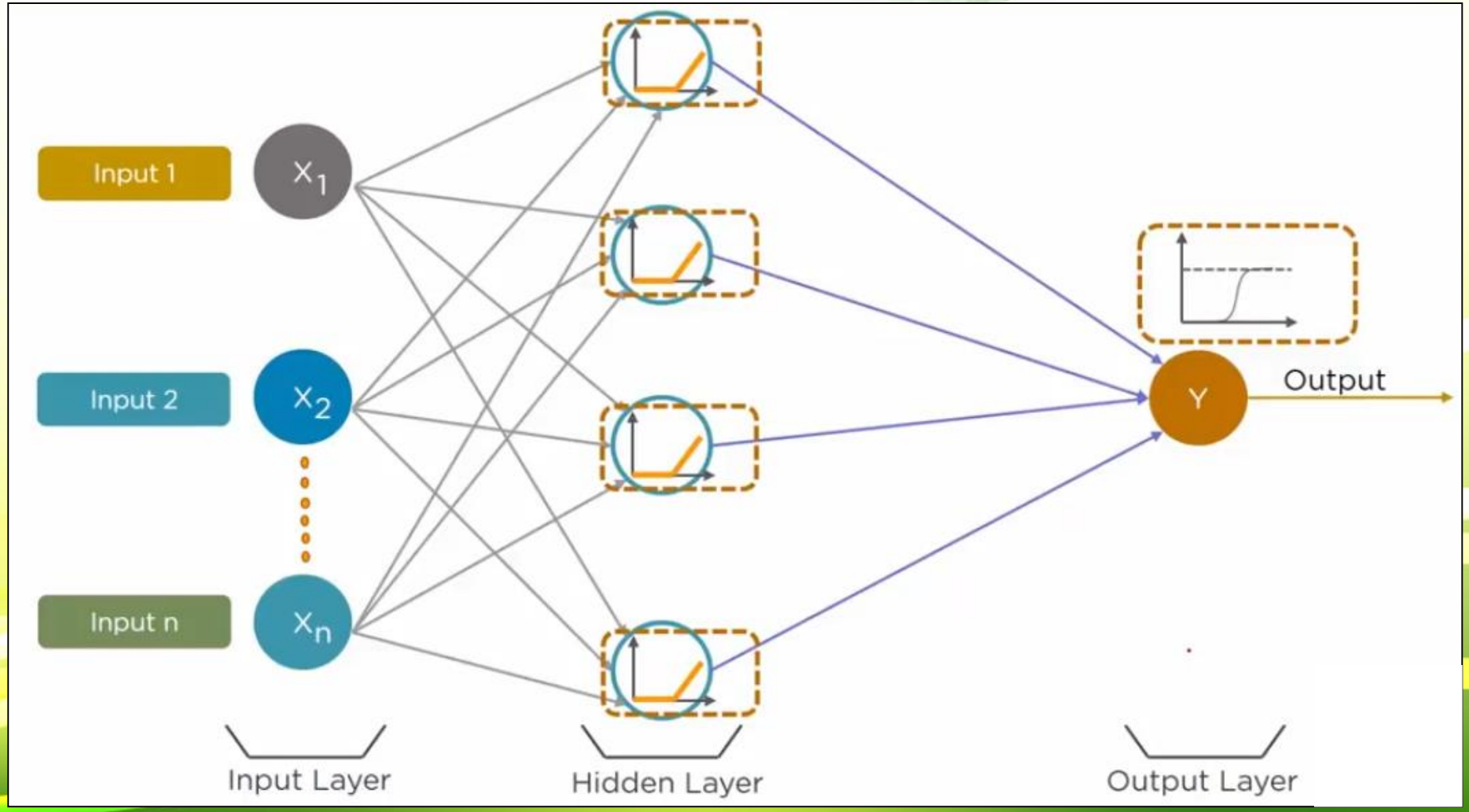
# Deep Learning ___ ANN __ Activation function

Hyperbolic Tangent function

This function is similar to Sigmoid function and is bound to range (-1, 1)



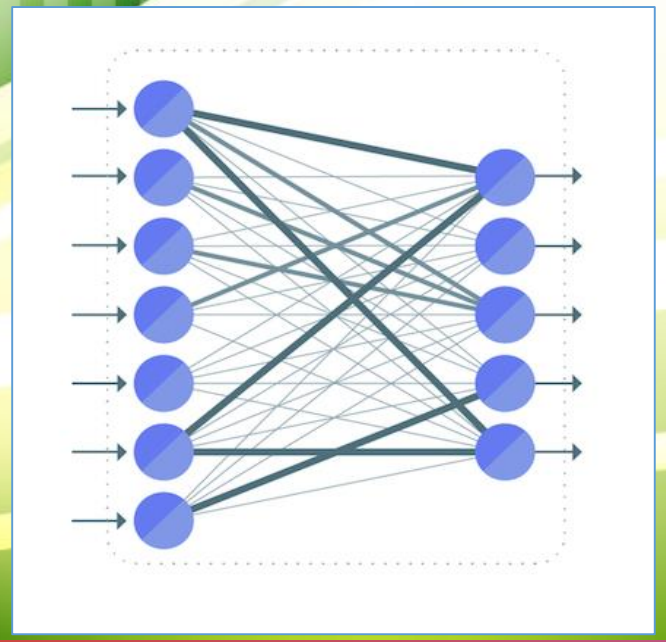
# Deep Learning ___ ANN __ Activation function



# Deep Learning __ ANN

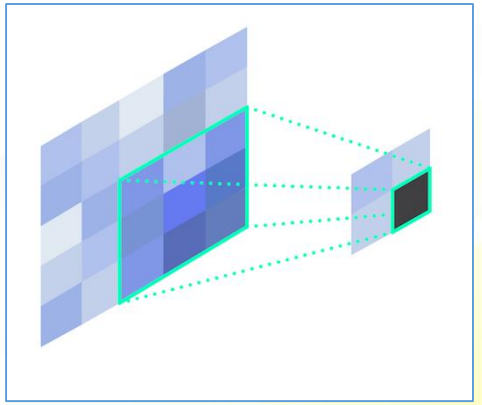
4 common types of NN layers: Fully connected, Convolution, Deconvolution, and Recurrent.

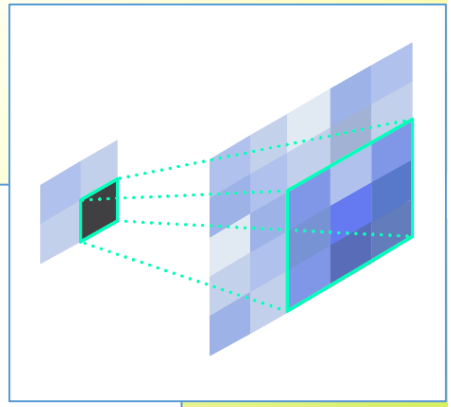
- **Fully connected layers** connect every neuron in one layer to every neuron in the next layer.
- Fully connected layers can become computationally expensive as their input grows, resulting in a combinatorial explosion of vector operations to be performed, and potentially poor scalability.

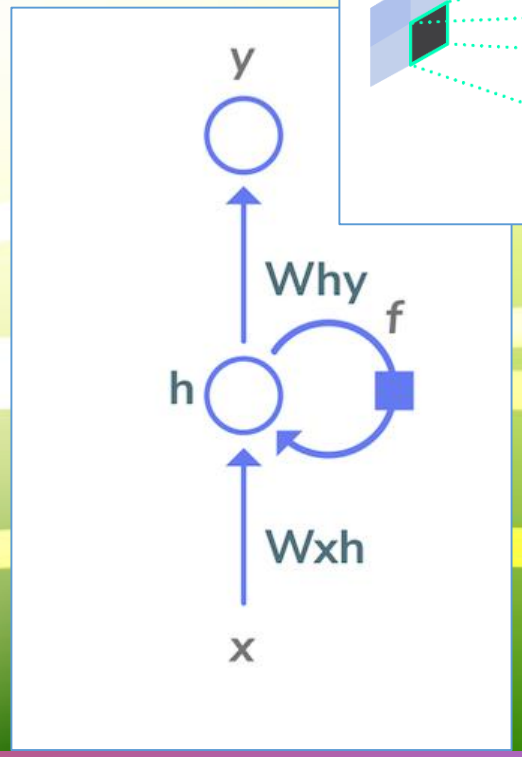


# Deep Learning __ ANN

- Convolution Layer is an important type of layer in a CNN.
  - Its most common use is for detecting features in images, in which it uses a filter to scan an image, a few pixels at a time, and outputs a feature map that classifies each feature found.
- Deconvolution Layer is a transposed convolution process that effectively upsamples data to a higher resolution.
- Recurrent Layer includes a "looping" capability such that its input consists of both the data to analyze as well as the output from a **previous calculation performed by that layer**.
  - Recurrent layers form basis of recurrent neural networks (RNNs).





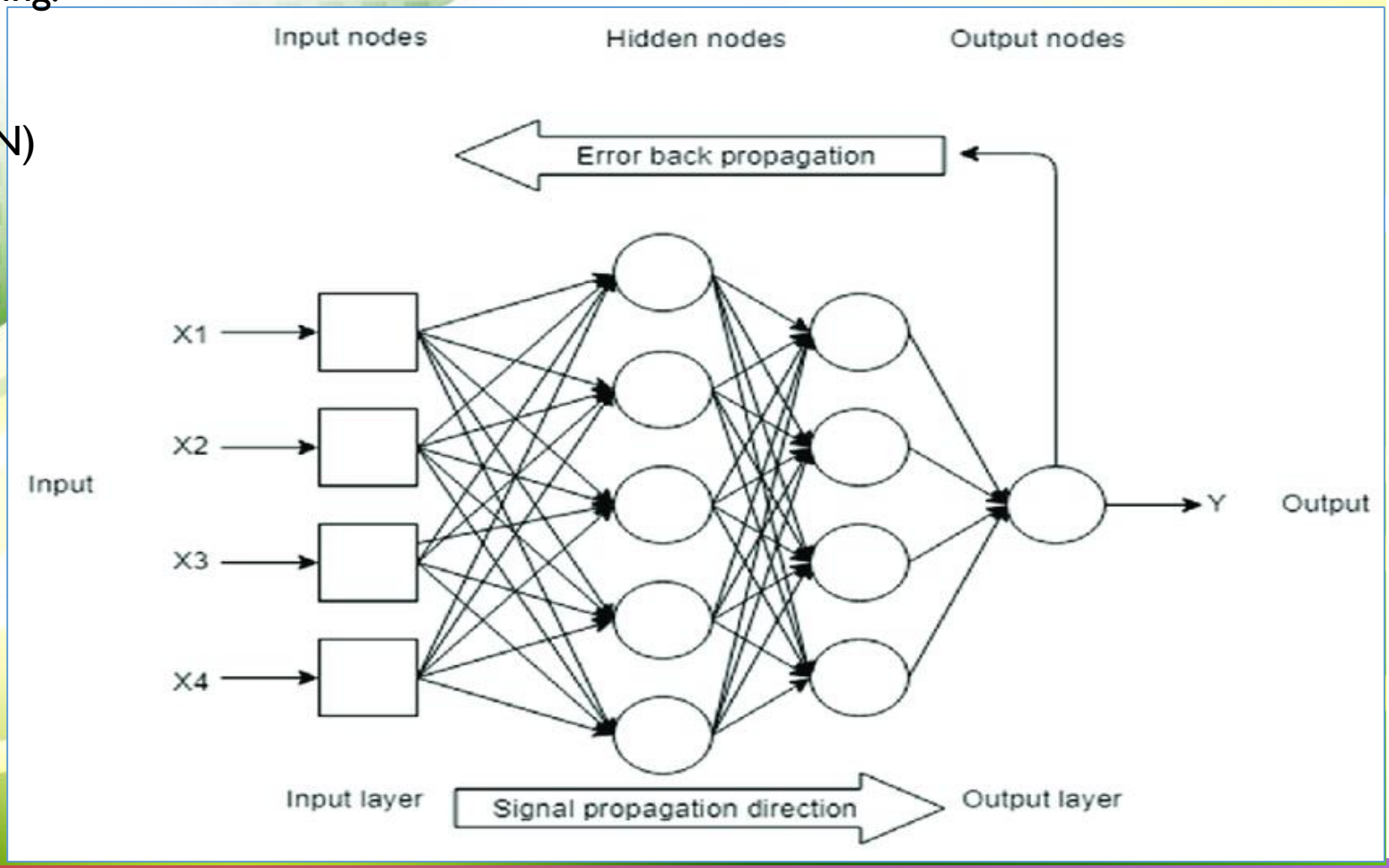


# Deep Learning __ ANN

Types of Neural Networks in Deep Learning:

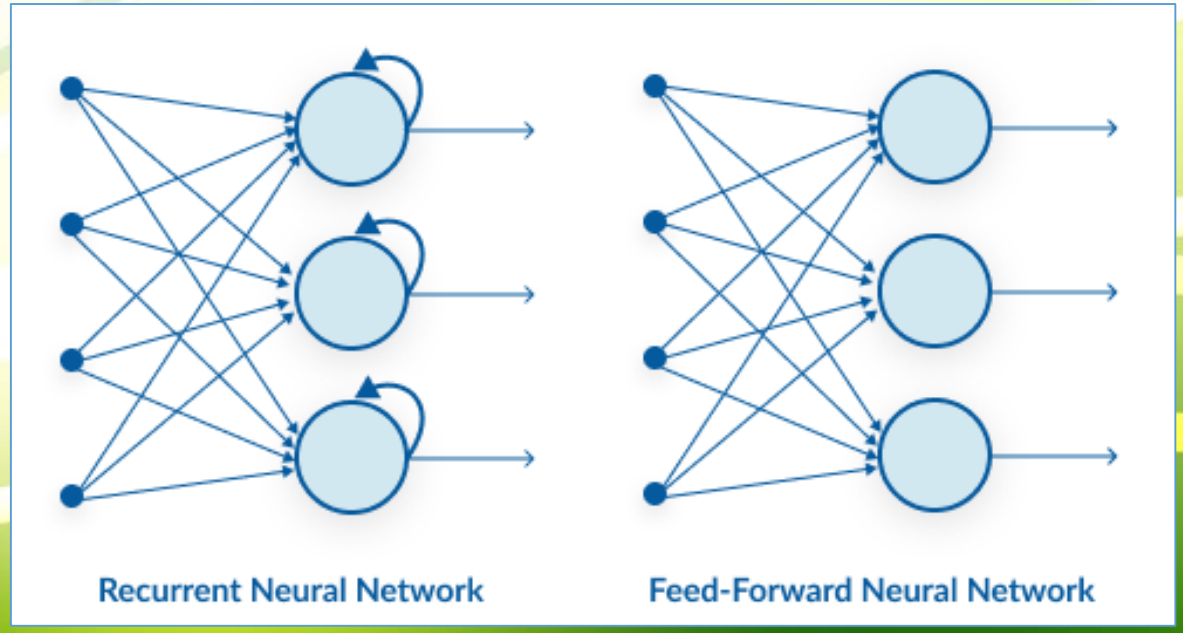
- Artificial Neural Networks (ANN)
- Convolution Neural Networks (CNN)
- Recurrent Neural Networks (RNN)

ANN is also known as a **Feed-Forward Neural network** because inputs are processed only in the forward direction.

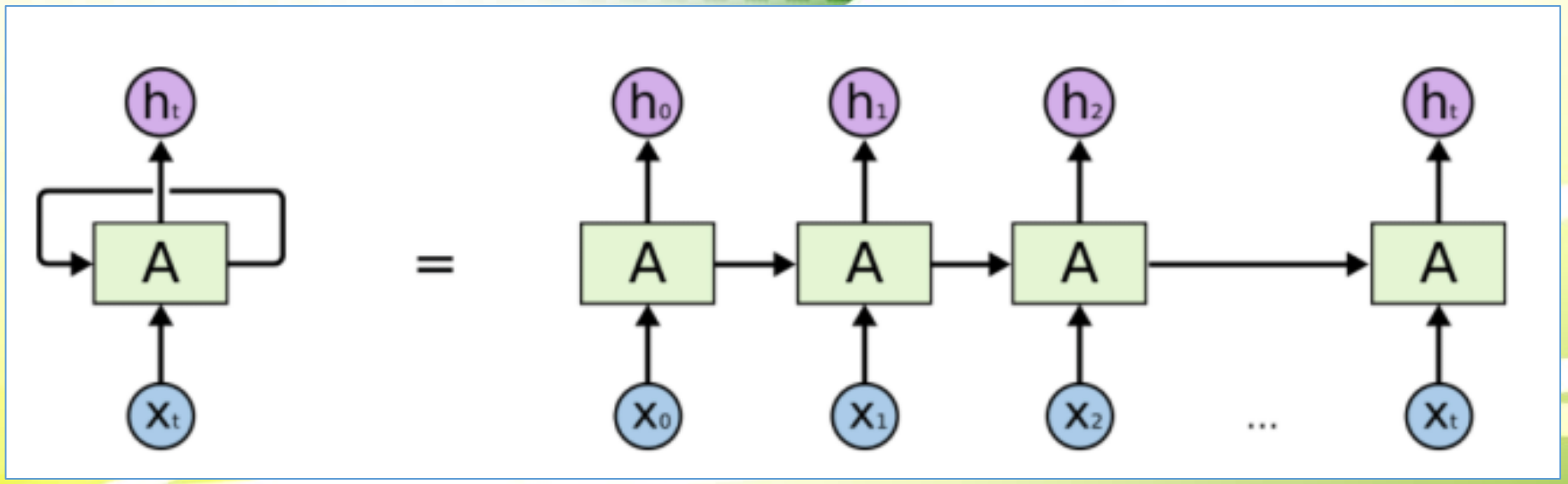


# Deep Learning __ RNN

- RNN has a recurrent connection on hidden state. This looping constraint ensures that sequential information is captured in input data.
- RNN is widely used to solve the problems related to: Time Series data, Text data, Audio data, etc.
- RNN captures the sequential information present in input data i.e. dependency between words in the text while making predictions.

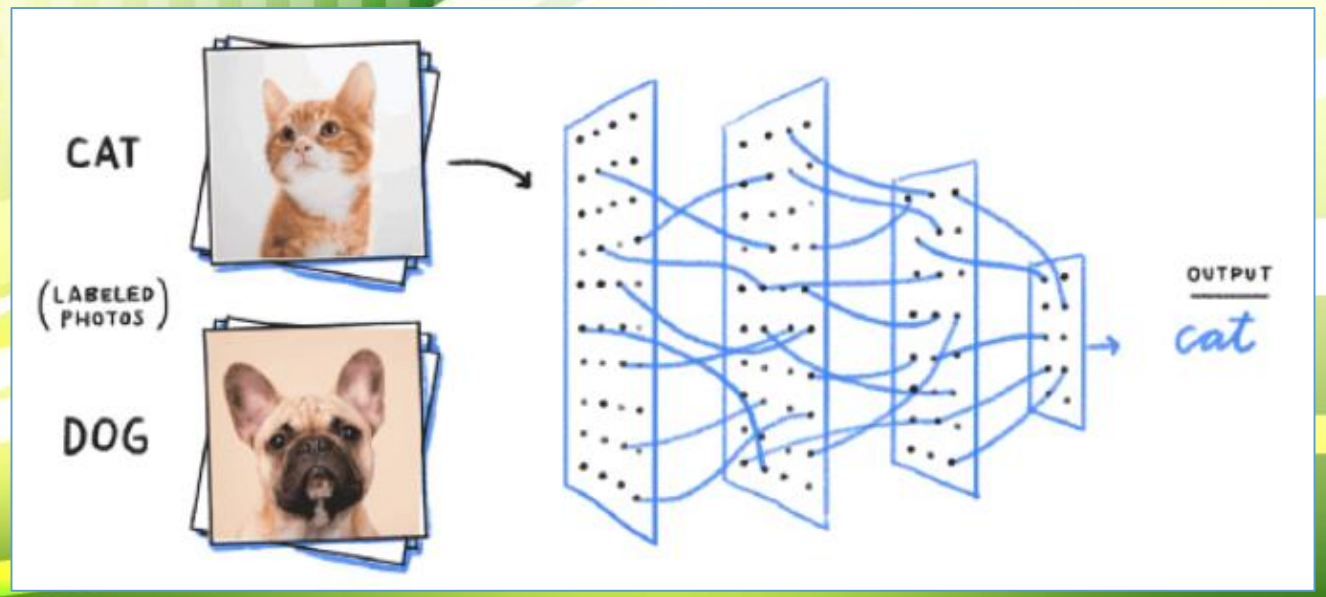


# Deep Learning __ RNN

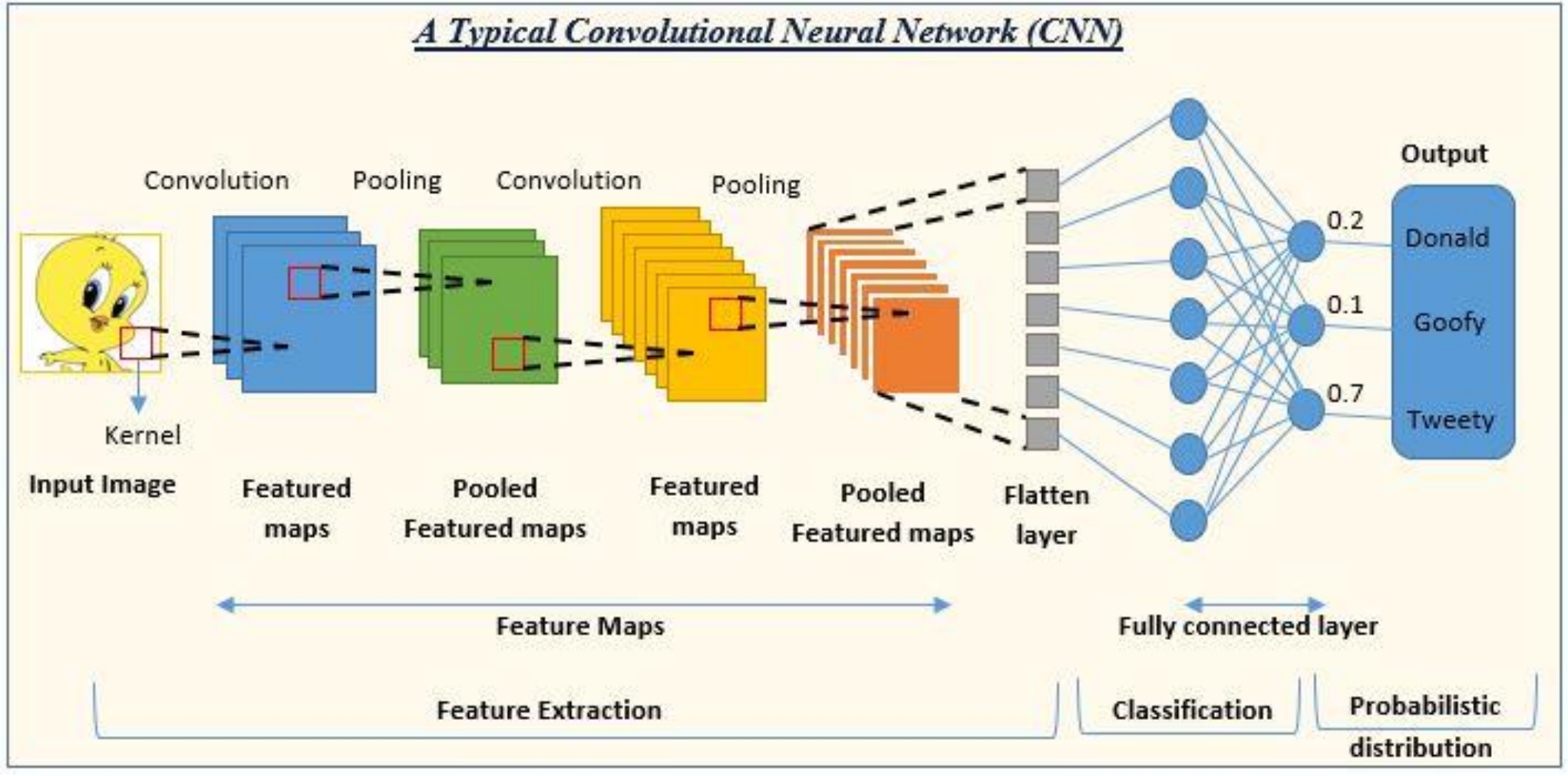


# Deep Learning __ CNN

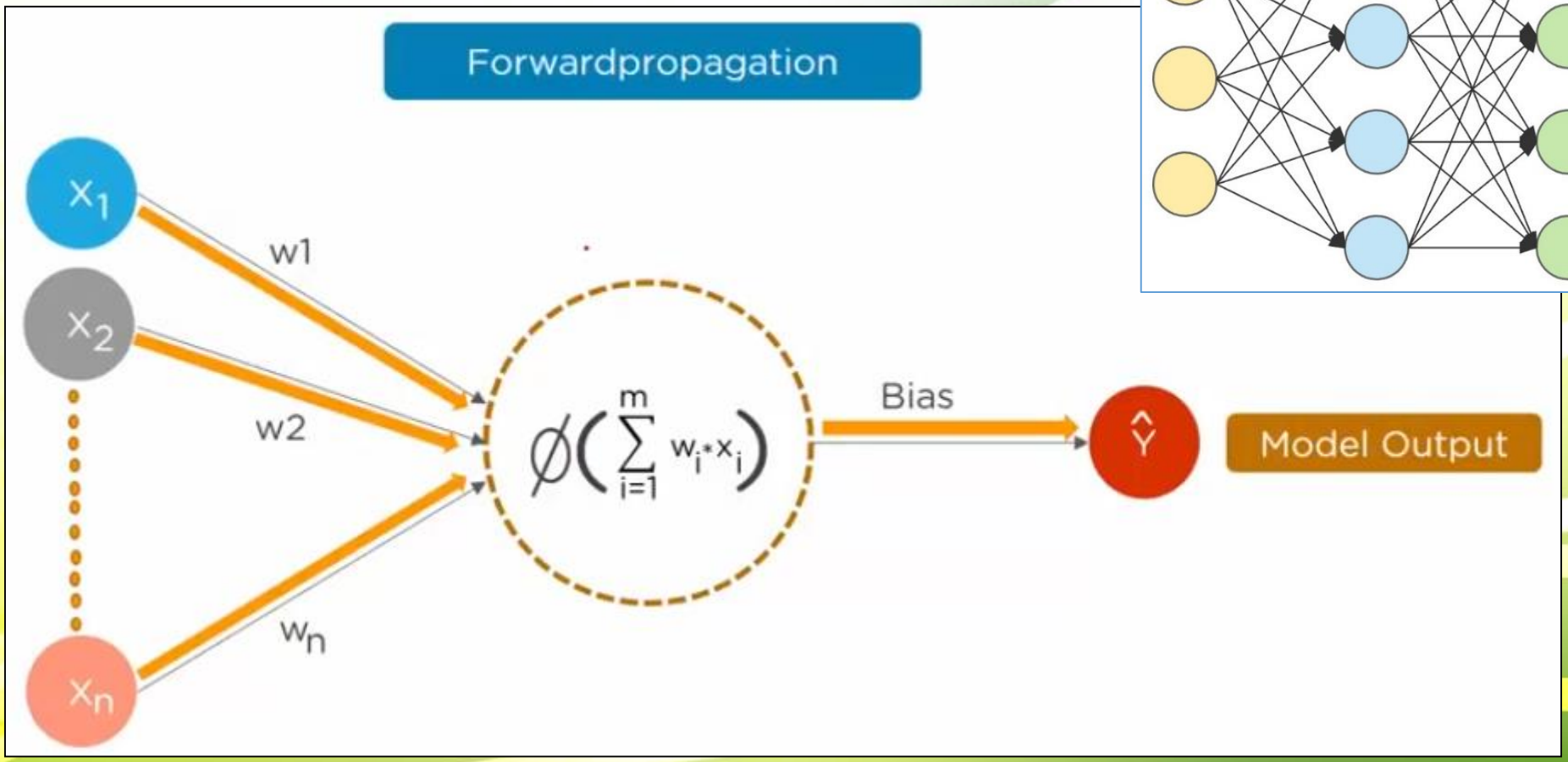
- Convolutional neural networks (CNN) models are being used across different applications and domains, especially in image and video processing projects.
- The building blocks of CNNs are filters (kernels).
- Kernels are used to extract relevant features from input using convolution operation.

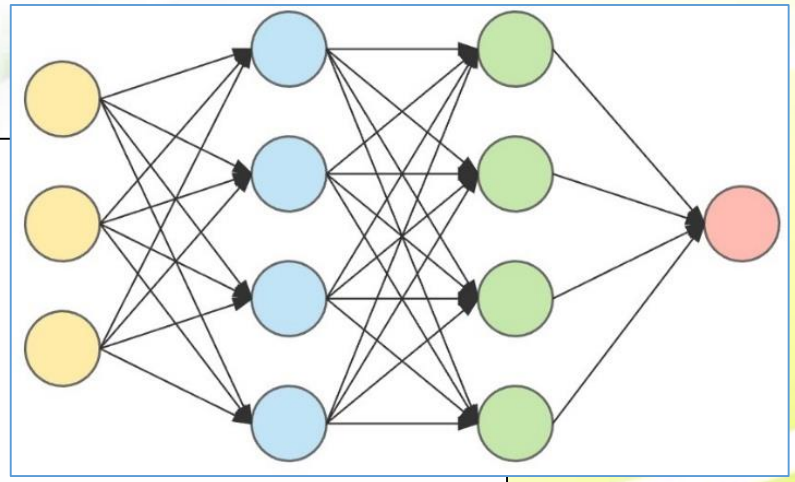


# Deep Learning __ CNN

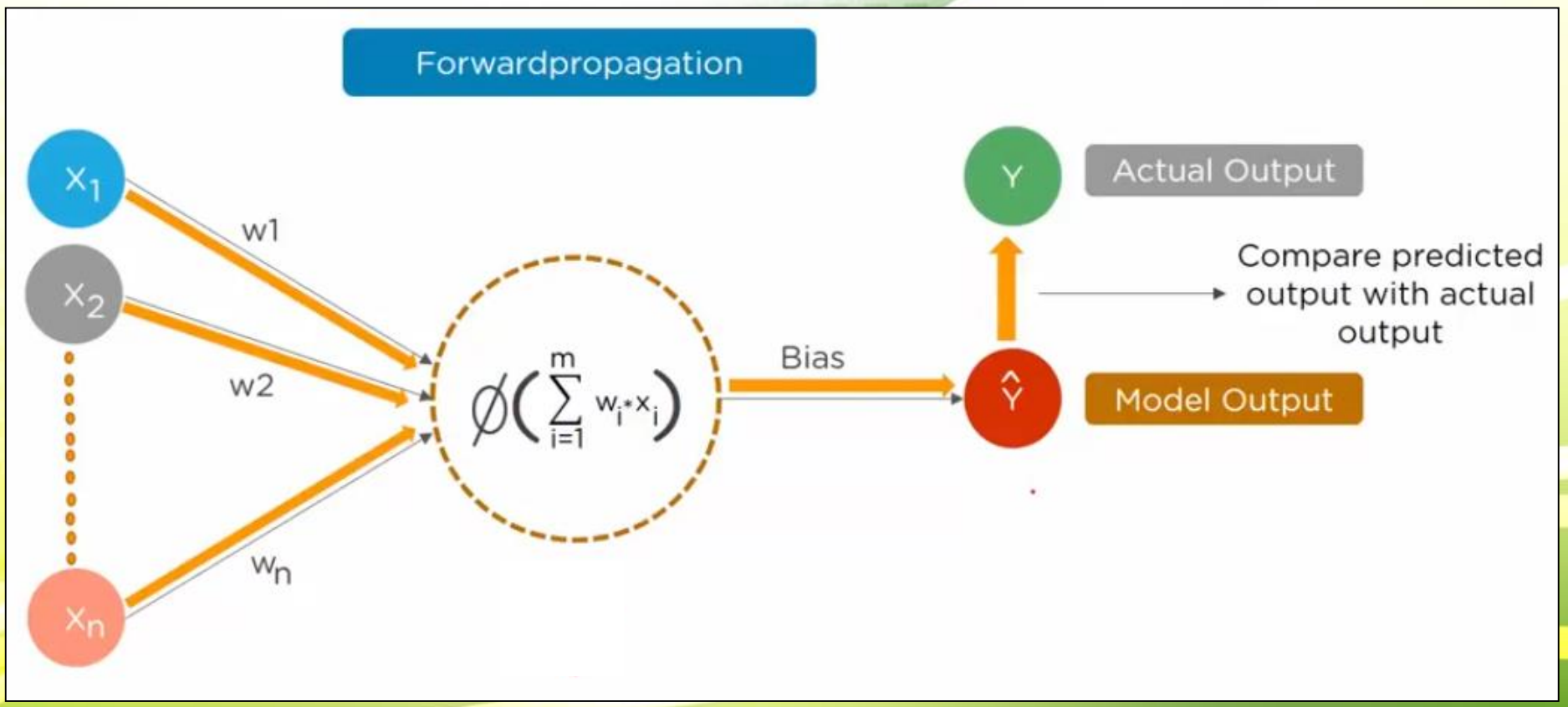


# Deep Learning __ ANN



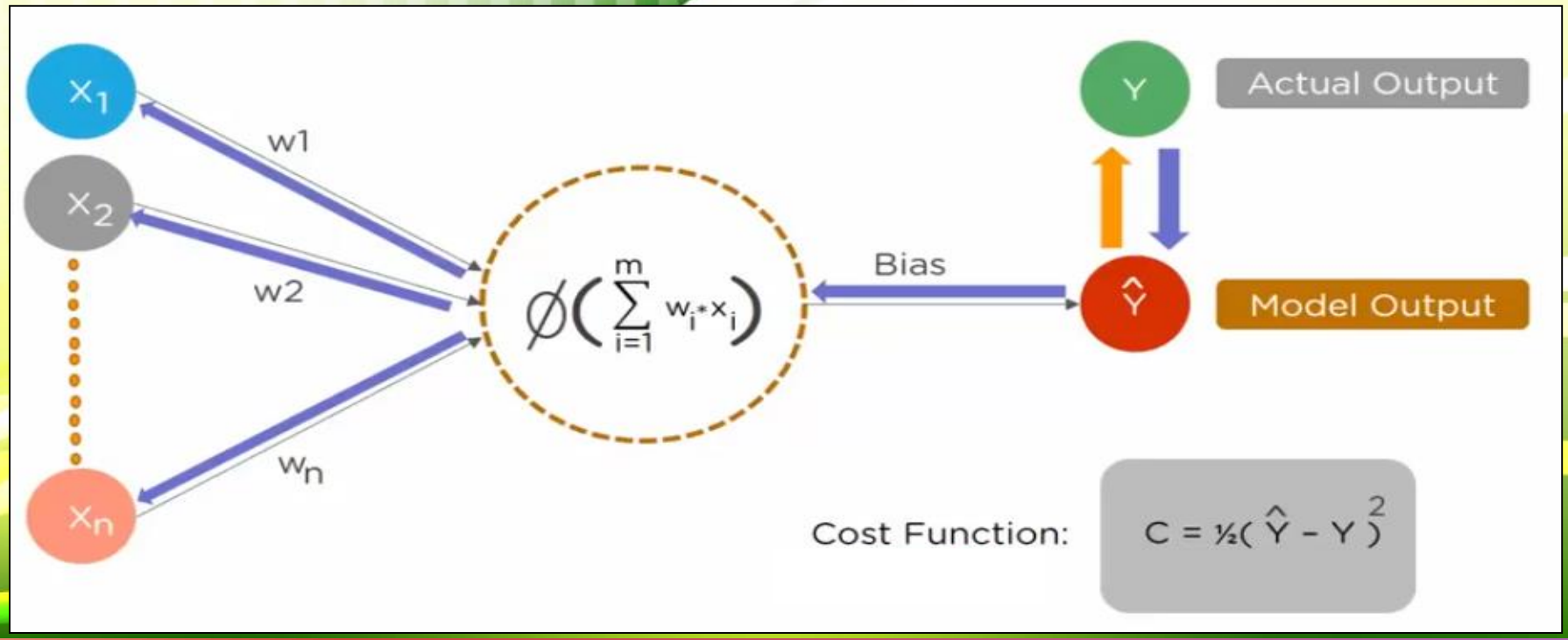



# Deep Learning __ ANN



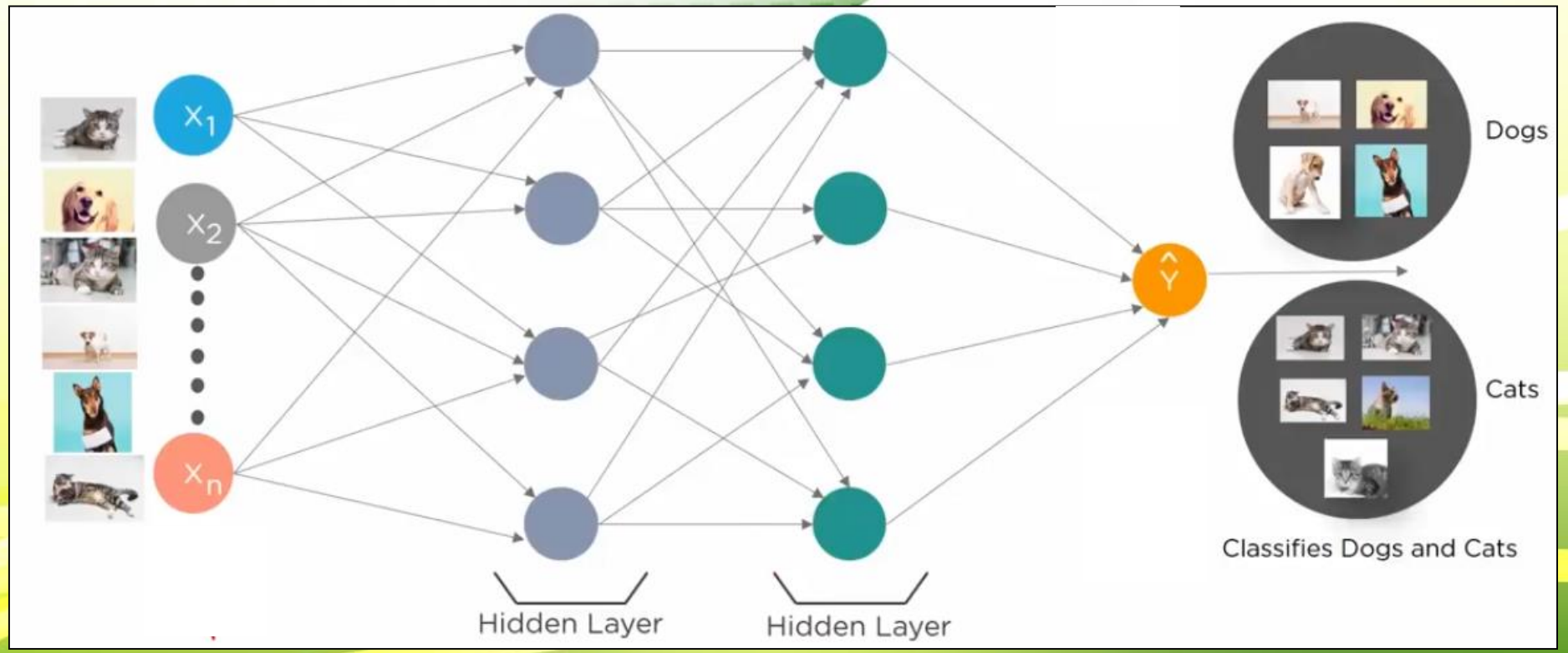
# Deep Learning __ ANN

- After Training ANN, ***BackPropagation*** method helps to improve performance of network.
- ***Cost function*** helps to reduce the error rate.
- Cost is difference between ANN predicted output and actual output (for Training Dataset).
- Cost is reduced by adjusting weights & biases iteratively throughout the Training process.

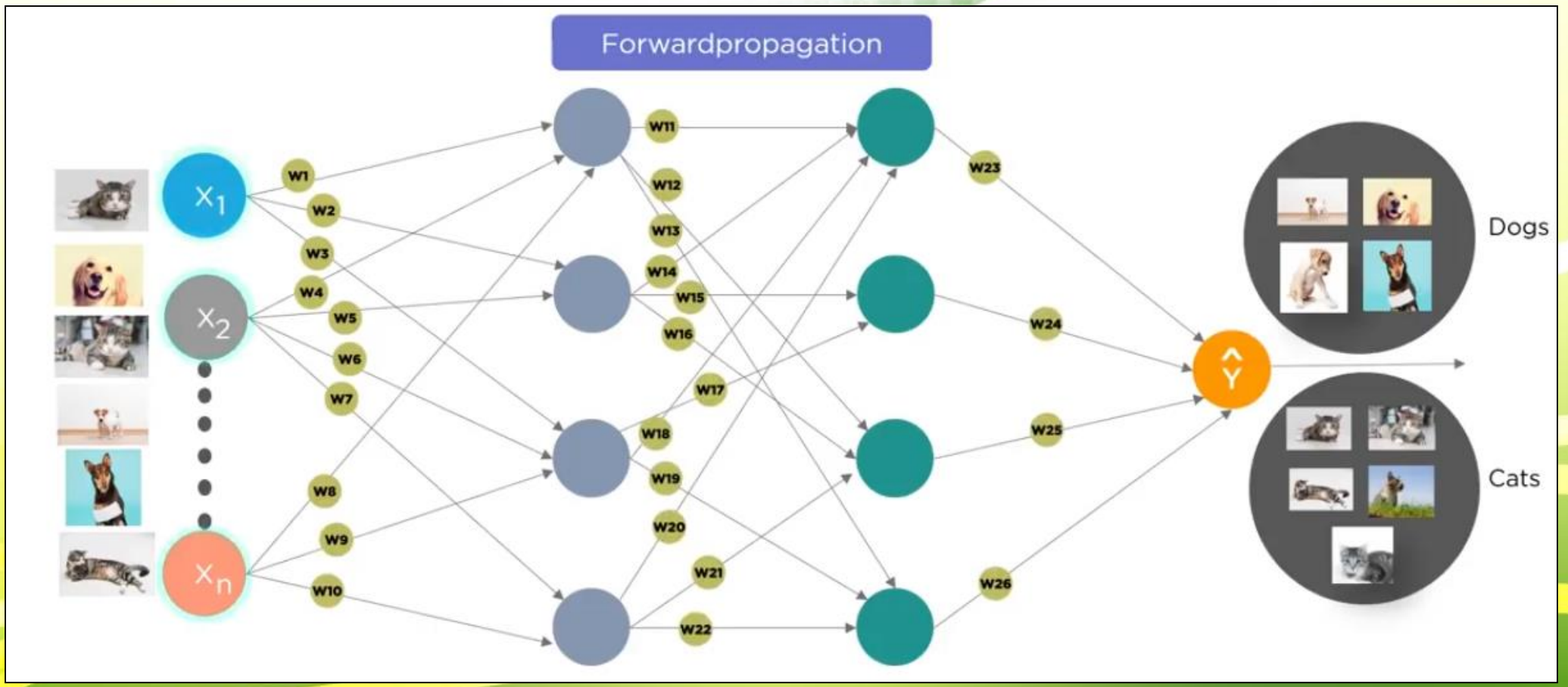


# Deep Learning __ ANN __ How it works

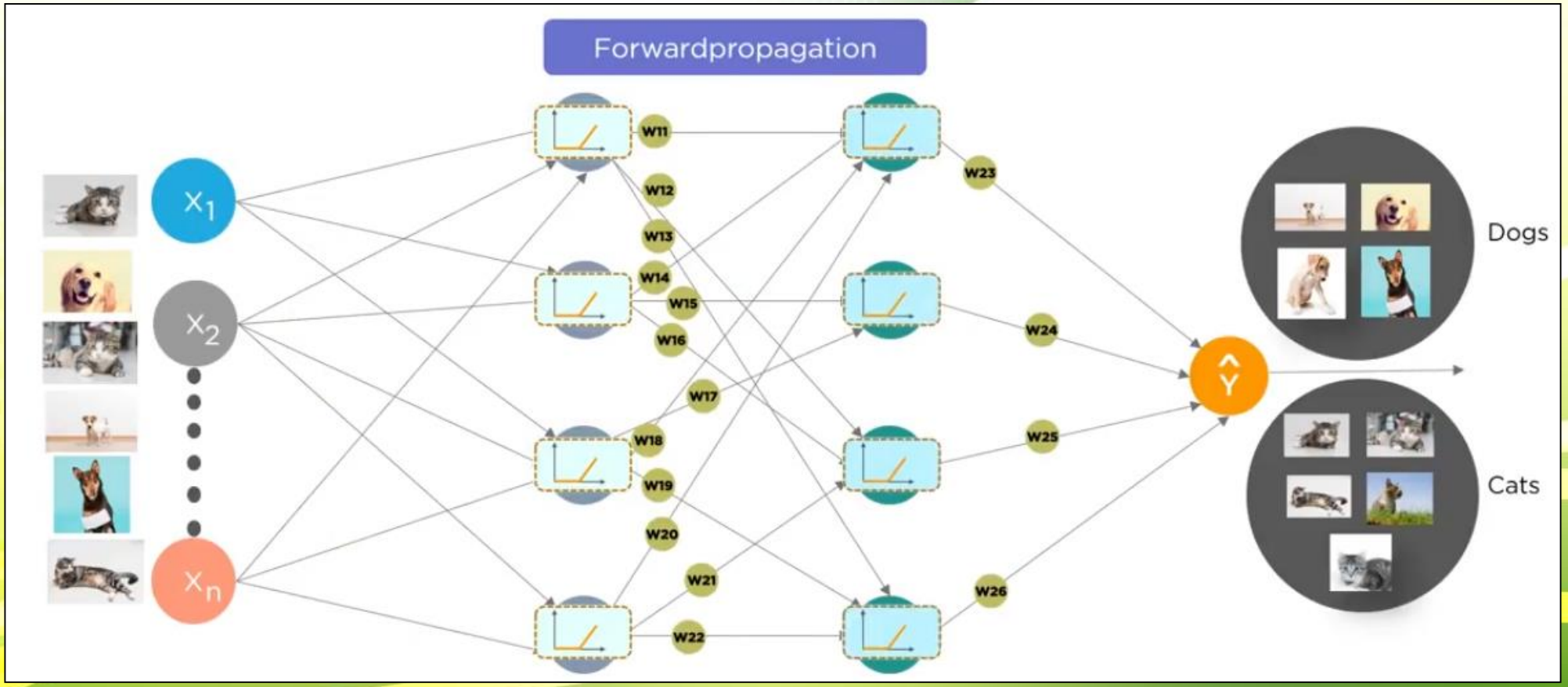
- Identify Dogs and Cats.



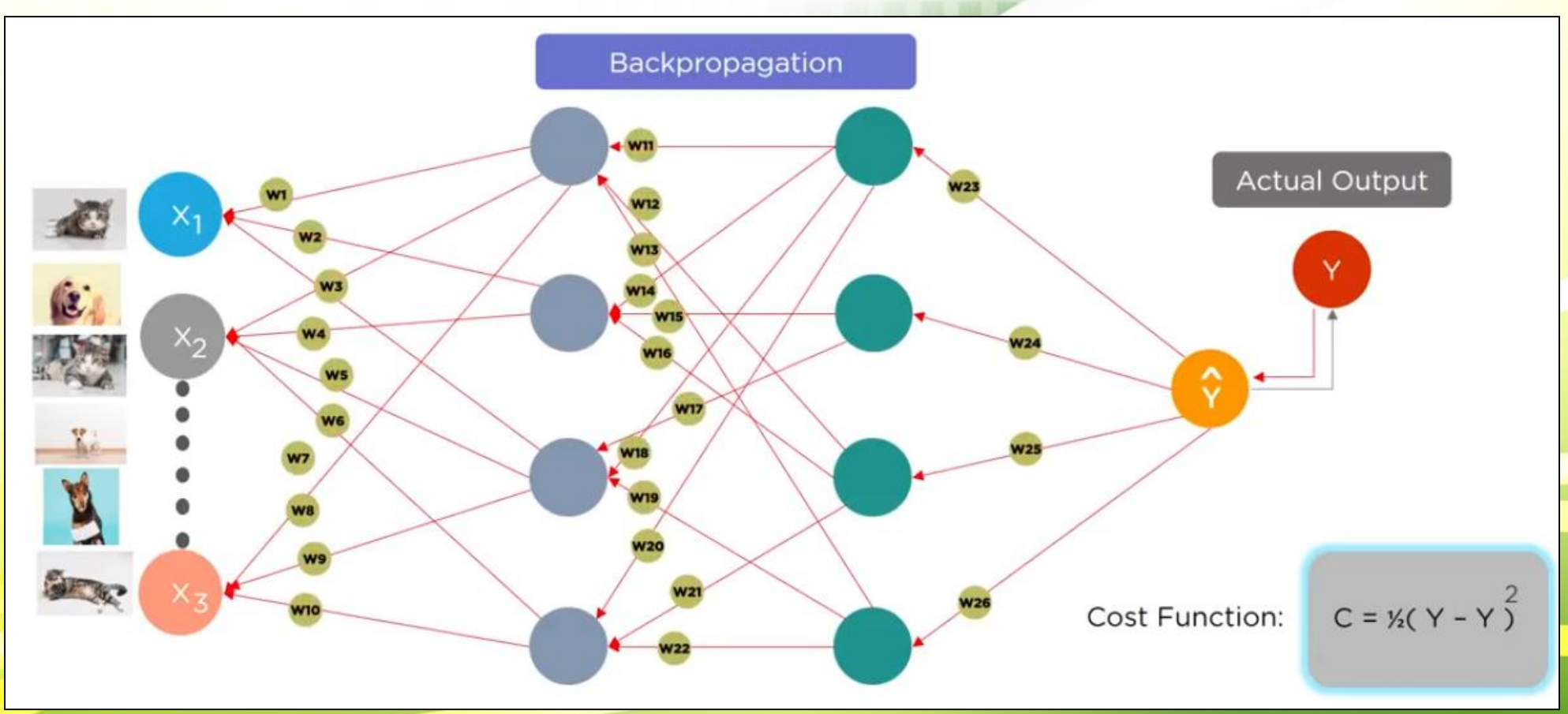
# Deep Learning __ ANN



# Deep Learning __ ANN



# Deep Learning __ ANN



# Hyperparameters

- **Parameters** are configuration variables that are internal to the model, and a **model learns them on its own**.
  - Example; Weights or Coefficients of independent variables in the Linear regression model, SVM, or weight, and biases of a neural network, cluster centroid in clustering.
  - They are used by the model for making predictions.
  - These are usually not set manually.
- Model **Hyperparameters** are those parameters that are explicitly defined by the user to control the learning process.
  - These are usually defined manually by the machine learning engineer.
  - One cannot know the exact best value for hyperparameters for given problem. The best value can be determined either by the rule of thumb or by trial and error.

# Hyperparameters

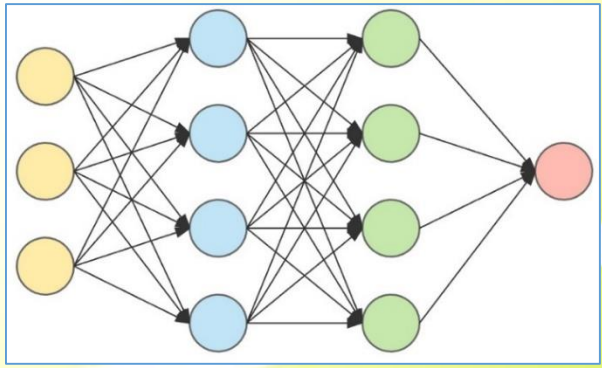
- Hyperparameters are used to improve model learning, and their values are set before starting the learning process of the model.
  - These parameters are explicitly defined by user to control the learning process.
  - These are external to the model, and their values cannot be changed during the training process.
- Some examples of Hyperparameters in Machine Learning
  - k in kNN or K-Nearest Neighbour algorithm
  - Train-test split ratio
  - Branches in Decision Tree
  - Number of clusters in Clustering Algorithm
  - Learning rate for training a neural network
  - Batch Size
  - Number of Epochs

# Hyperparameters

- Broadly hyperparameters can be divided into two categories, which are given below:
  - **Hyperparameter for Optimization:**
    - Process of selecting the best hyperparameters to use is known as hyperparameter tuning/optimization.
    - *Example;* ***Learning Rate, Batch size.***
  - **Hyperparameter for Specific Models:**
    - Hyperparameters that are involved in the structure of model are known as hyperparameters for specific models.
    - *Example;* ***Number of Hidden Units, number of layers.***

# Hyperparameters

- **Number of Layers**: A neural network is made up of vertically arranged components, called layers (input layers, hidden layers, and output layers).
  - A higher-layered neural network gives a better performance than a lower-layered network.
  - Model complexity may increase with higher-layered NN.
  - For some specific Neural network, a lower number of layers make a better model.



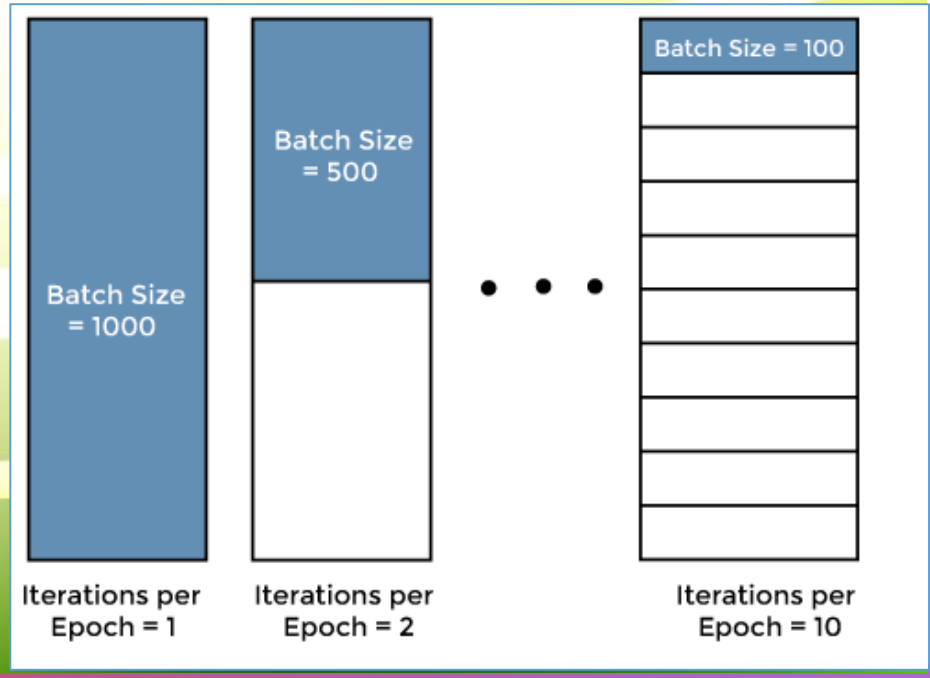
- **Number of Hidden Units**: Hidden units refer to components comprising layers of processors between input and output units in a neural network.
  - It is important to specify the number of hidden units hyperparameter for neural network.
  - It should be between the size of the input layer and the size of the output layer.
  - More specifically, the number of hidden units should be 2/3 of the size of the input layer, plus the size of the output layer.

# Hyperparameters

- **Learning Rate**: Controls how much model needs to change in response to estimated error for each time when model's weights are updated.
  - Selecting the optimized learning rate is a challenging task.
  - Low learning rate may slow down training process; and high learning rate may not optimize model properly.
- **Batch Size**: To enhance speed of learning process, training set is divided into different subsets (batch).
- **Number of Epochs**: An epoch can be defined as the complete cycle for training the machine learning model.
  - Epoch represents an iterative learning process.
  - Number of epochs varies from model to model, and most models are created with more than one epoch.

# Hyperparameters - EPOCH

- **Epoch** refers to one complete pass of the training dataset through the algorithm.
  - It takes a few epochs while training a machine learning model.
  - It will face an issue while feeding a bunch of training data in the model; due to limitations of computer storage.
    - To overcome this issue, training data is broken into small batches according to the computer memory or storage capacity.
    - *When all batches are fed exactly once to train the model, then this entire procedure is known as Epoch in Machine Learning.*
- *Example;*
  - *Total number of training examples = 3000;*
  - *Assume each batch size = 500;*
  - *Total number of Iterations = Total number of training examples/Individual batch size*
  - *Total number of iterations = 3000/500 = 6;*
  - ***1 Epoch = 6 Iterations.***



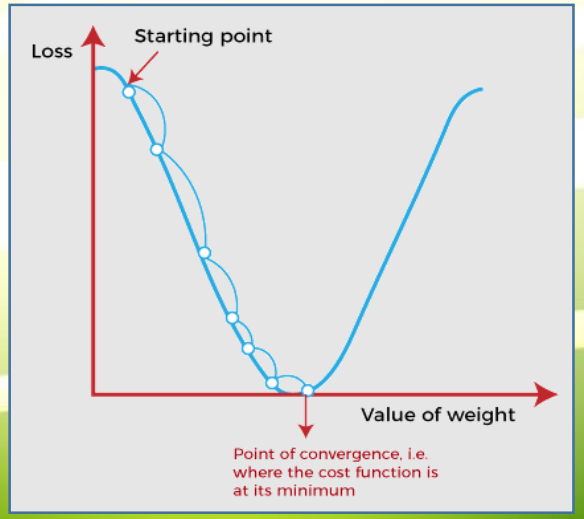
# Hyperparameters - EPOCH

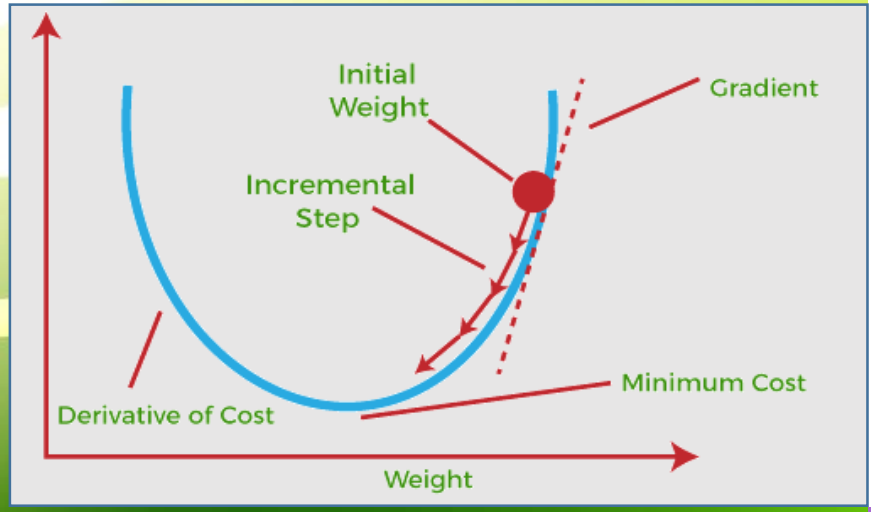
- Model with a single epoch may lead to overfitting in the model.
- Training a model typically requires multiple numbers of Epochs.
- Better generalization can be achieved with new inputs by using more epochs in the training of machine learning model.
- Given the complexity and variety of data in real-world applications, hundreds to thousands of epochs may be required to achieve reasonable test data correctness.
    - To determine the right number of epochs, a validation error is taken into account.
    - The number of epochs is increased until there is a reduction in a validation error.
    - If there is no improvement in reduction error for the consecutive epochs, then it indicates to stop increasing the number of epochs.

# Gradient Descent

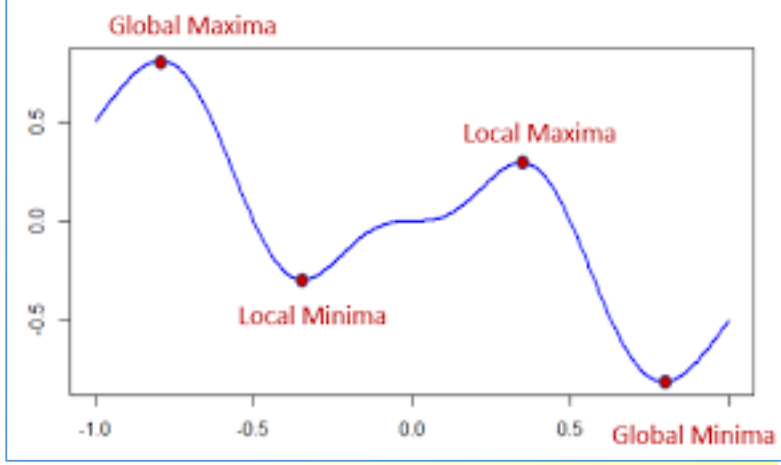
- *Gradient Descent is an optimization algorithm which is used for optimizing the cost function or error in the model.*
  - Enables the models to take the gradient or direction to reduce errors by reaching to least possible error.
  - Here direction refers to how model parameters should be corrected to further reduce the cost function.
- *It helps in finding the local minimum of a function.*
- Its an iterative process where the model gradually converges towards a minimum value, and if the model iterates further than this point, it produces little/zero changes in the loss.
  - This point is known as **convergence**, and at this point, the error is least, and the cost function is optimized.

- Main objective of gradient descent is to minimize cost function (error between expected & actual).
- To minimize the cost function, two factors are required: **Direction & Learning Rate**



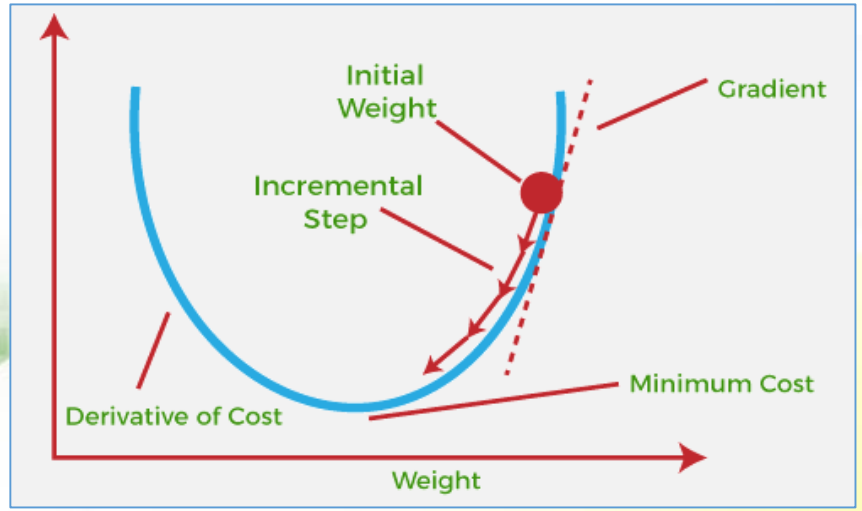
# Gradient Descent




**Challenges with the Gradient Descent**

- **Local Minima and Saddle Point:**
  - Gradient descent aims to find global minimum; but sometimes it may fall trapped with saddle point and local minimum.
  - Local minima generate shape similar to global minimum, where slope of the cost function increases on both sides of the current points.
  - With saddle points, negative gradient only occurs on one side of the point, which reaches a local maximum on one side and local minimum on other side.
  - The name of local minima is because value of loss function is minimum at that point in a local region.
  - In contrast, the global minima value of loss function is minimum globally across entire domain loss function.

# Gradient Descent




**Challenges with the Gradient Descent**

- **Vanishing and Exploding Gradient**
  - Vanishing Gradient occurs when the gradient is smaller than expected.
    - During backpropagation, this gradient becomes smaller causing decrease in learning rate of earlier layers than later layer of the network.
  - Exploding gradient is just opposite to the vanishing gradient as it occurs when Gradient is too large and creates an unstable model.
  - These problems can be solved using the dimensionality reduction technique, which helps to minimize complexity within the model.

# There's NO End.