Deep Learning

Augmented reality technology has existed mostly as a Smartphone novelty --see yelp's "monocle" feature on their mobile app or the Layer augmented reality browser-- but the technology may soon become more integrated into our daily lives through the use of media-enhanced glasses. Most visibly, Google seems to be making the move toward augmented reality glasses. The New York Times recently reported that "The glasses will have a low-resolution built-in camera that will be able to monitor the world in real time and overlay information about locations, surrounding buildings and friends who might be nearby, according to the Google employees."

Traditional ML Algorithm

- Suppose we have n=100 features

- For Non-Linear Logistic Regression, a quadratic function of order 2 will have features = n*(n+1)/2 = **5050** features

$$x_1^2 + x_1x_2 + x_1x_3 + x_1x_4 + x_1x_5 + \ldots\ldots x_{99}x_{100} + x_{100}^2$$

- For order 3 it will be = n*(n+1)*(n+2)/6 = **1,71,700** features

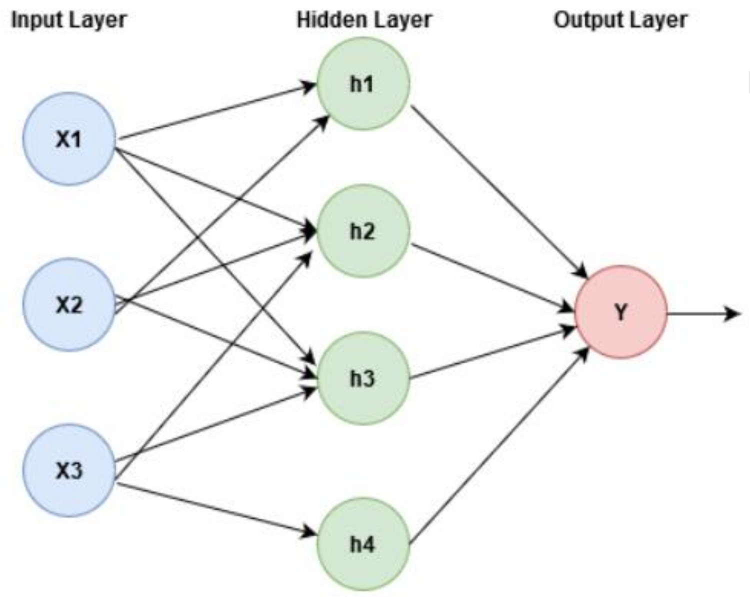
Traditional ML Algorithm

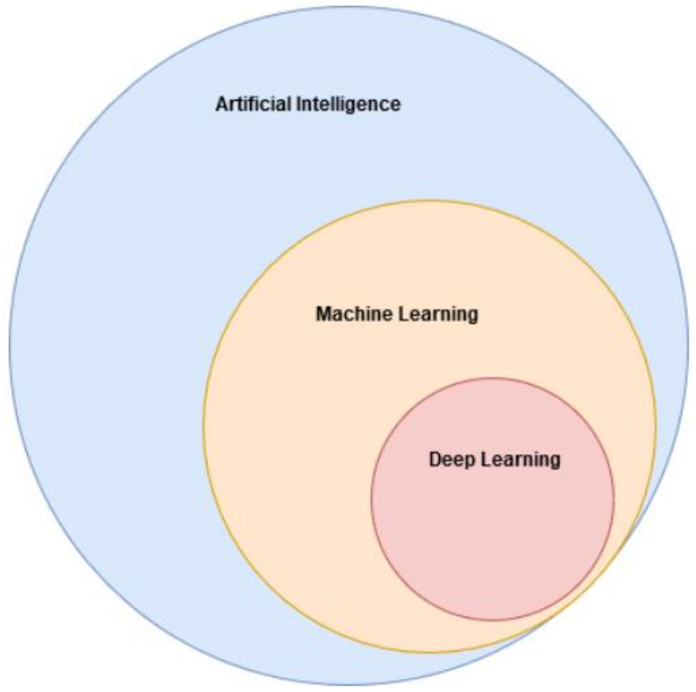
- Suppose We have an Image of pixel size 100 * 100
- Total pixels = 100 * 100 = 10,000 pixels, n = **10,000** (30,000 for RGB)
- n = 10,000 for order 2 quadratic feature will have total **100 Million features** (900 Million feature for RGB)

Deep Learning

- Deep learning is part of a broader family of machine learning methods based on artificial neural networks.
- The “deep” in deep learning refers

to the depth of the network.

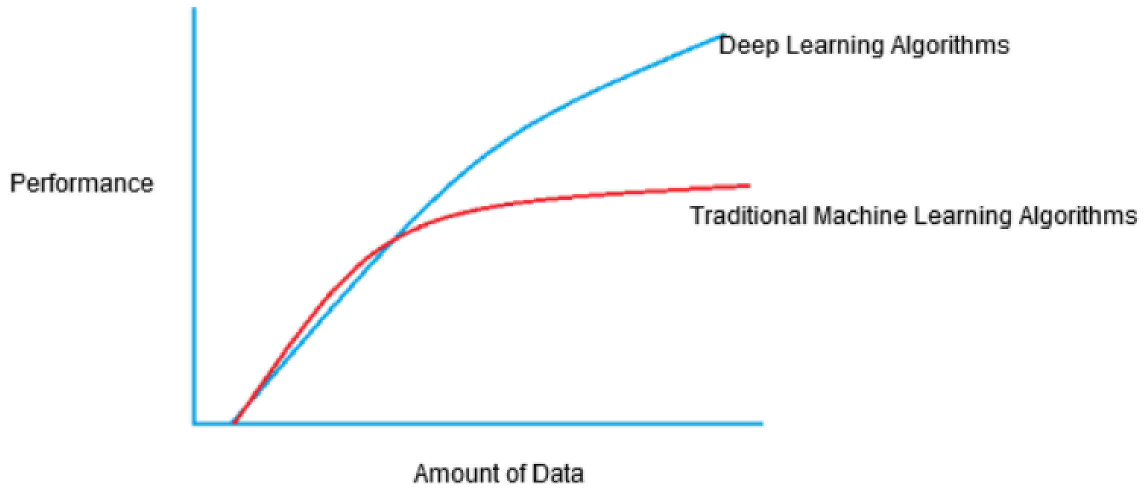




Deep Learning Performance

- Performance Increases as amount of data increase

- Need large data to learn



Application

- Self driving cars
- Automatic hand writing generation
- Face Detection (facebook)
- Fraud Detection
- Language translation

Recap

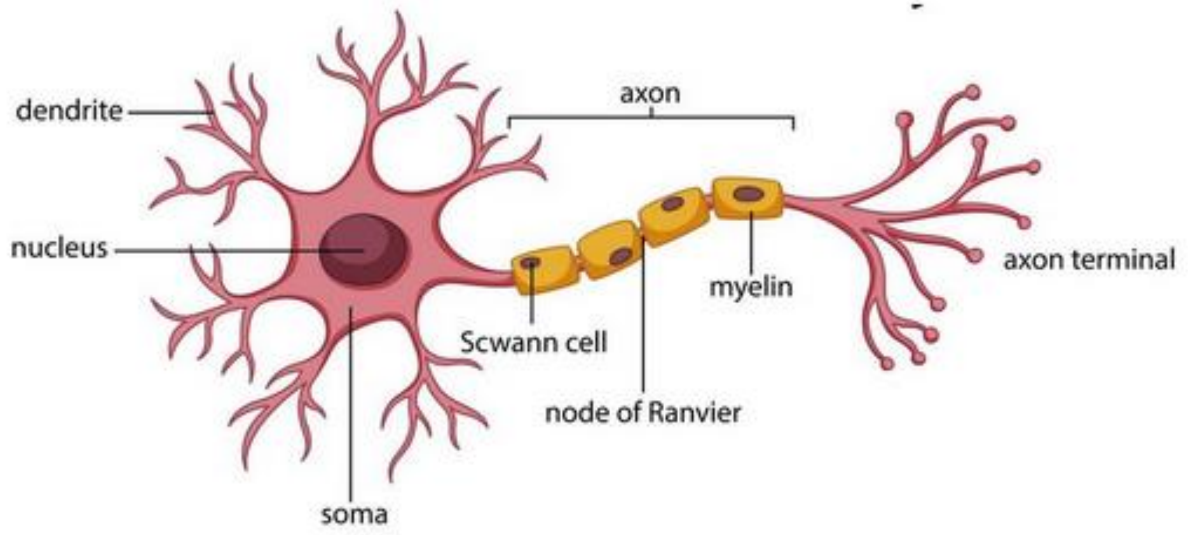
- Limitation of Traditional ML Algorithm
- Deep Learning
- Performance Difference
- Application

Artificial Neural Network

- Implementation of Deep Learning
- Inspired by biological systems

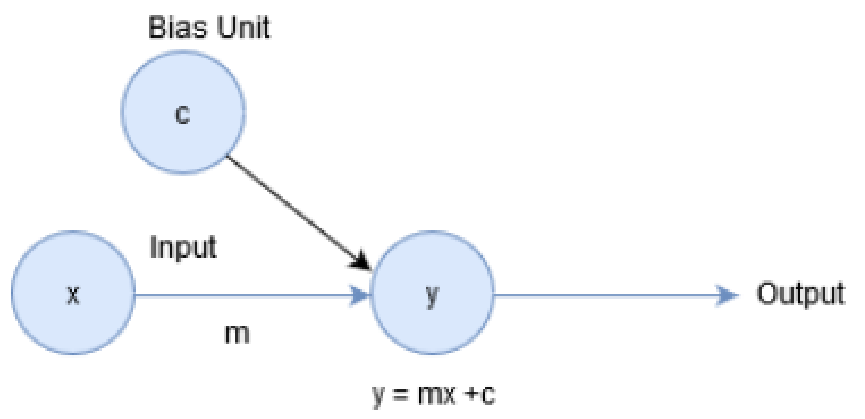
- Dendrites are the structures on the neuron that receive electrical messages, to process these signals, and to transfer the information to the soma of the neuron

- Axons: primary transmission lines of the nervous system



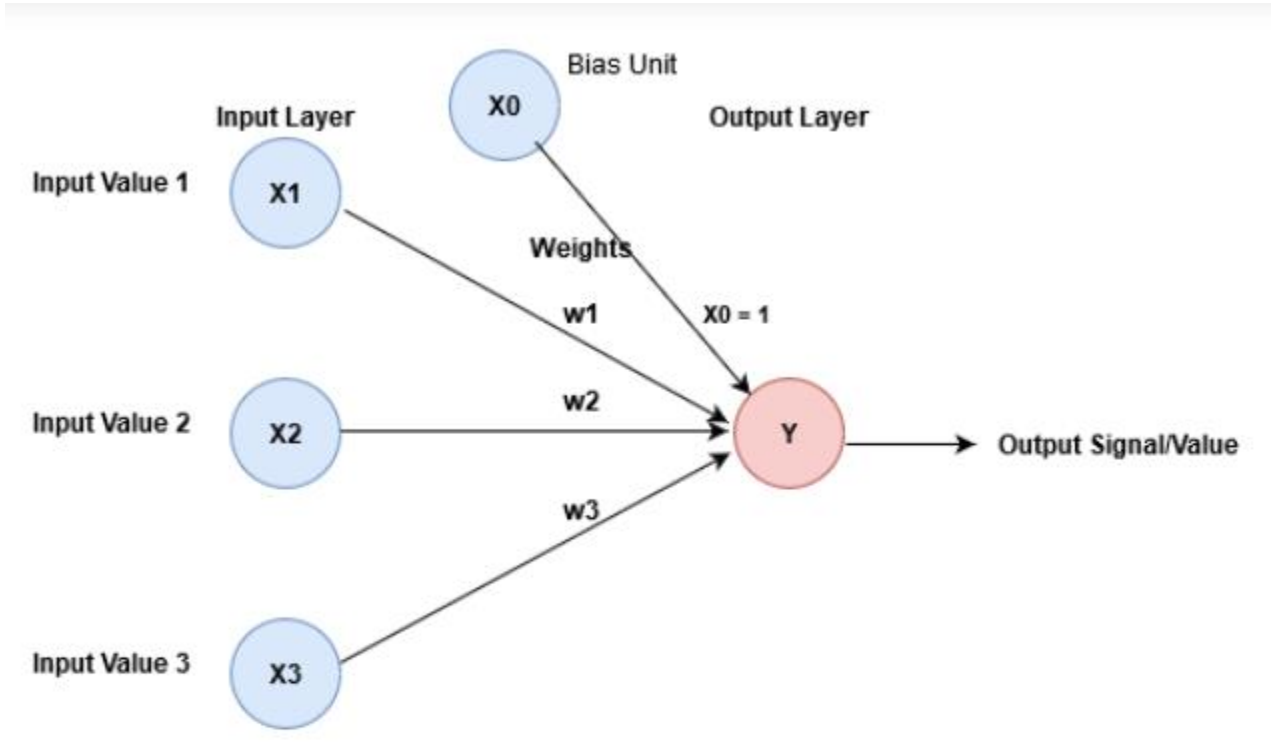
How Neural Network Works?

- Y = mx +c
- One Input
- One Bias/constant Unit: Its a constant which helps the model in a way that it can fit best for the given data



How Neural Network Works?

- Multiple Input

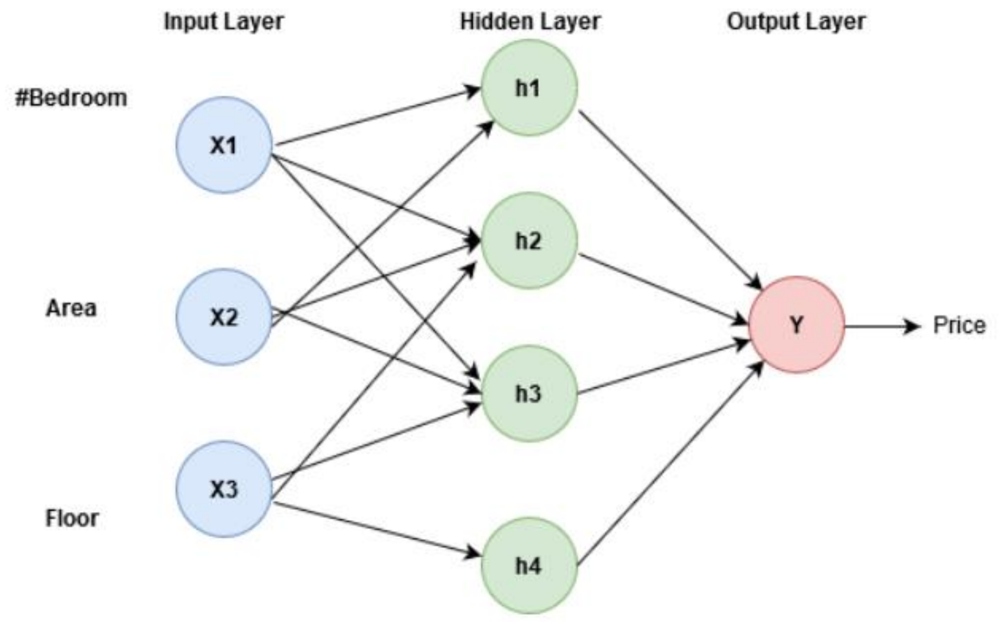


- Output = x0 + x1*w1 + x2*w2 + x3w3
- Weight determines how much influence an input will have in the next Neuron

Neural Network with Hidden Layer

- Why name Hidden: Intermediate values can't see in training set

- Network learned using feature x1, x2 & x3 and created own feature h1,h2,h3,h4.



Selecting Hidden Layer

- Hidden layers equals one

- # of Neurons (Hidden Layer) =

  Mean(Neurons in Input + Output Layer)

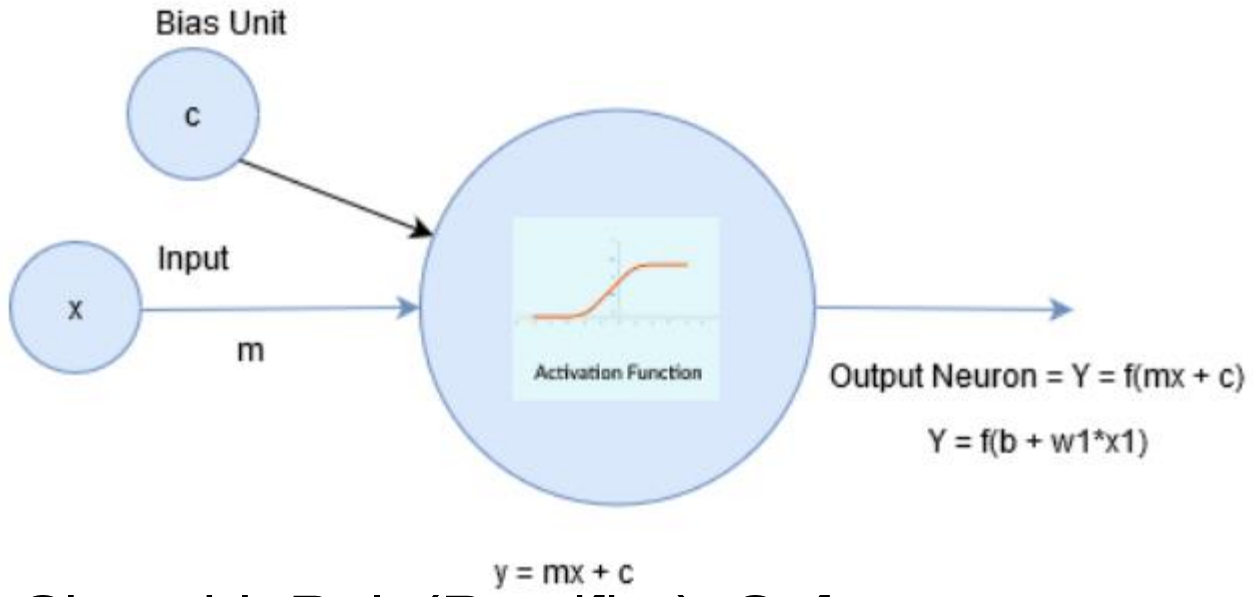
Hidden Layers

- Human See “flying bird with sun”. No separation

- Algorithms try to detect the local intensity variations such as: a key point, an edge, etc.
- Objective of Object Detection Problems: Minimising this gap between high level representations (interpreted by humans) and low level features (detected by algorithms)
- DNN use several hidden layers to hierarchically learn high level representation of the image. For instance, first layer might detect edges in the image, second layer might detect curves present in the bird, third layer might detect the whole bird, etc. They try to bridge the gap between high level representation and low level features.

Activation Function

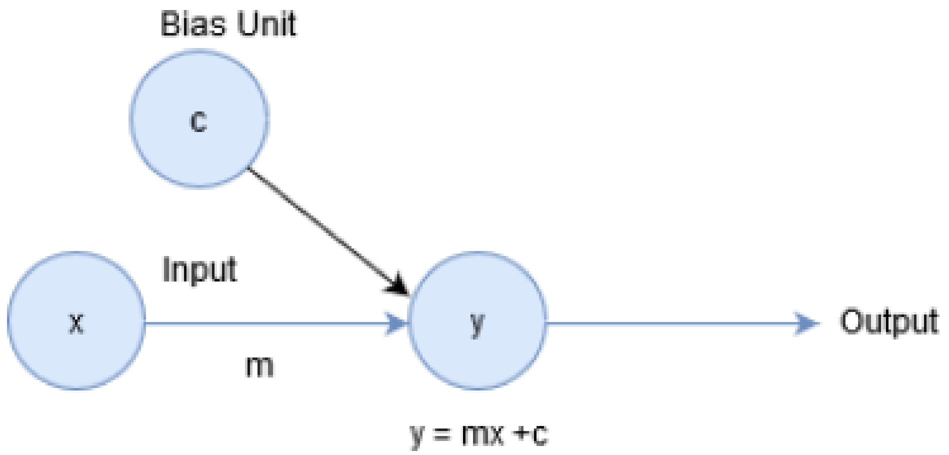
- determines whether it should be activated or not, based on whether each neuron's input is relevant for the model's prediction.
- It helps to standardize the output of each neuron.



- E.g: Threshold, Sigmoid, Relu(Rectifier), Softmax

Single Layer Perceptron

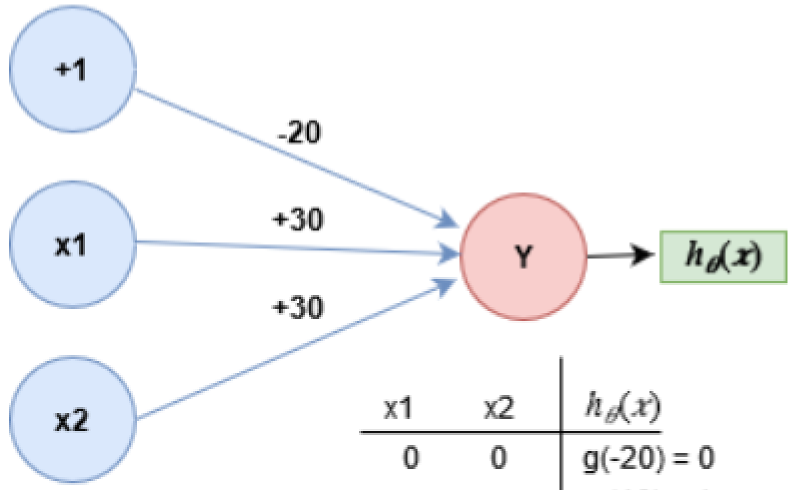
- neuron along with a set of input nodes connected to the output via weighted edges, is a perceptron, the simplest neural network.



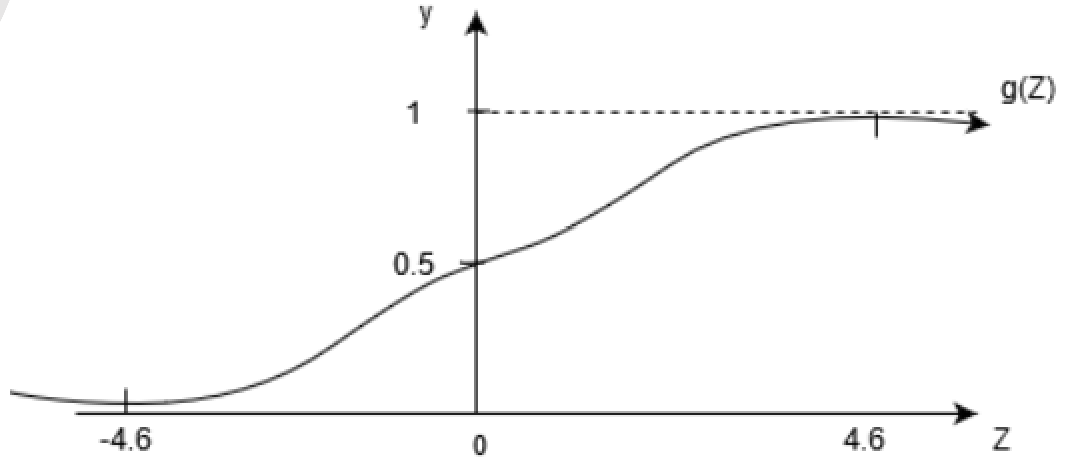
Implement OR Gate

$$h_\theta(x) = g(-20 + 30x_1 + 30x_2)$$

OR Gate



x1	x2	$h_\theta(x)$
0	0	g(-20) = 0
0	1	g(10) = 1
1	0	g(10) = 1
1	1	g(40) = 1



Algorithm to Implement Simplest NN

1. **Initialize Learning rate, bias and weights**

2. **Function perceptron: perceptron(x_1, x_2, output)**

- Takes input variable and actual output
- Calculate Error = ½*(actual - predicted)^2
- Recalculate the weights

weights = weights + error * input * lr

3. Function predict: predict(x_1, x_2)

- Takes input variable and actual output
- Predict = Calculate Output

Algorithm to Implement Simplest NN Cont…

4. Call perceptron for each row of OR gate

5. Run in Loop for multiple times to train the Network

6. Take Input values from user to predict the value

Different Activation Functions

Linear Function:

- Using Linear function only will make the output layer to be a linear function as well so we can't map a non-linear dataset

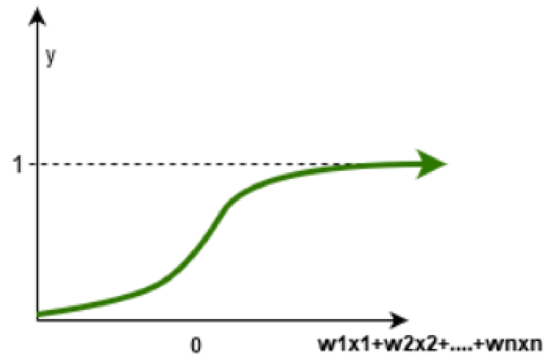
Step Function:

- we define threshold values and have discrete output values
- if(z > threshold) — "activate" the node (value 1)
- if(z < threshold) — don't "activate" the node (value 0)
- So, we have value either 0 or 1
- issue here is that it is possible multiple output classes/nodes to be activated (to have the value 1). So we are not able to properly classify/decide.

Sigmoid Function:

- It is a non-linear function
- Value range is (0,1)
- classify values either 1 or 0

$$\theta(x) = \frac{1}{1+e^{-x}}$$



Forward and Backward Propagation

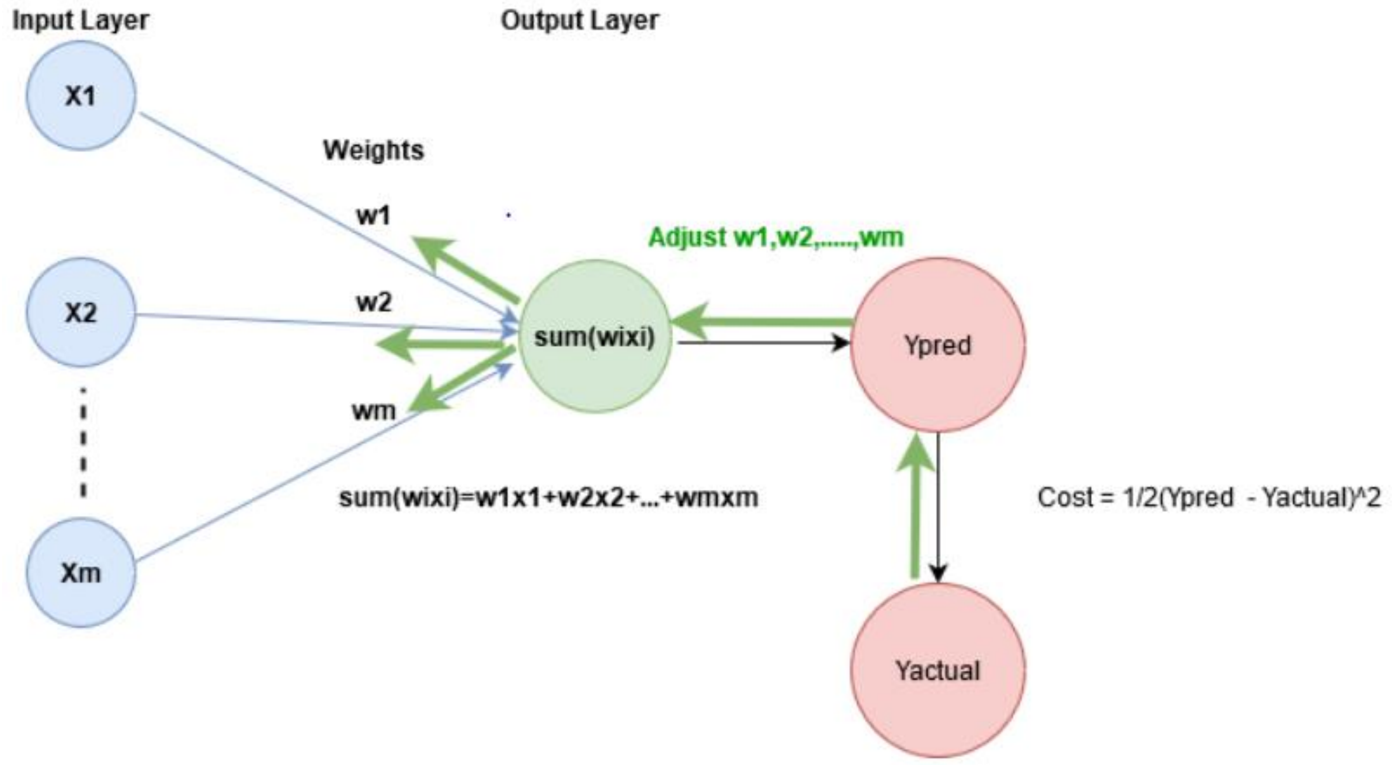
- Cost reduces with adjustment in weight(w)

- Error propagates from right to left and update the weights according to how much they are responsible for the error.

- Determining how changing the weights impact the overall cost in the neural network.

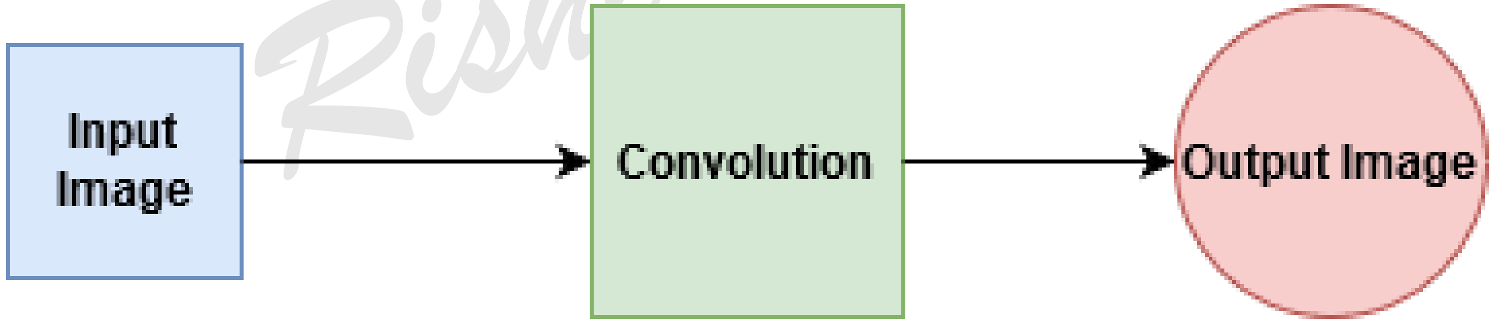
- The **Learning rate** decides by how much we update the weights

weight = weight + Error*Lr*input



Convolutional Neural Network(CNN)

- Same as Ordinary NN because these are also made of Neurons
- These Neurons can learn biases and weights
- Data is fed into Ordinary NN after converting to 1-D array
- If data is Image than it will be an issue
- CNN takes the 2-D structure of the Images while processing them

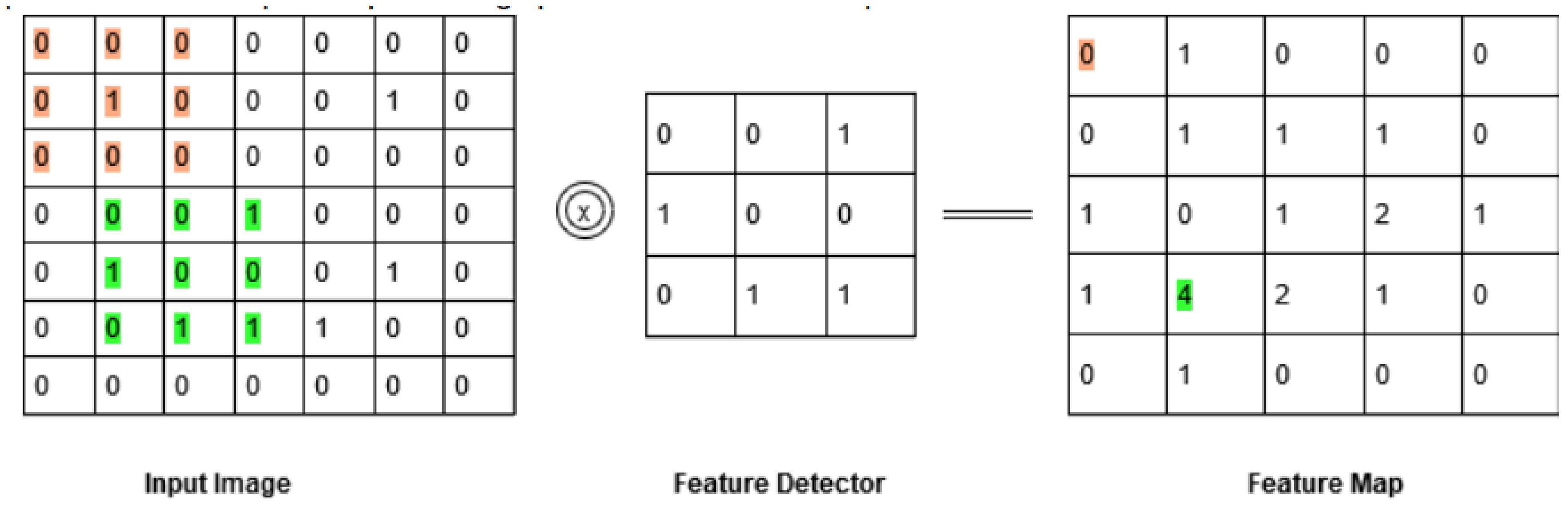


Steps In CNN

- $CONVOLUTION \longrightarrow MAXPOOLING \longrightarrow FLATTENING \longrightarrow FULLCONNECTION$

CONVOLUTION

- Identify features and put it in feature map while preserving special relation between pixels



0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input Image

0	0	1
1	0	0
0	1	1

Feature Detector

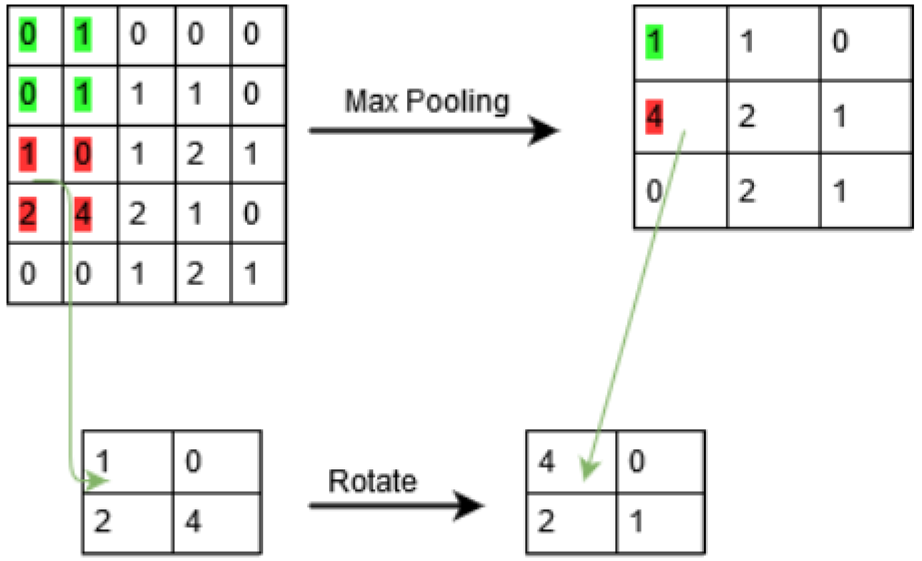
0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	1	0	0	0

Feature Map

- We create many Feature Maps by taking different Feature Detector to obtain our first convolution layer

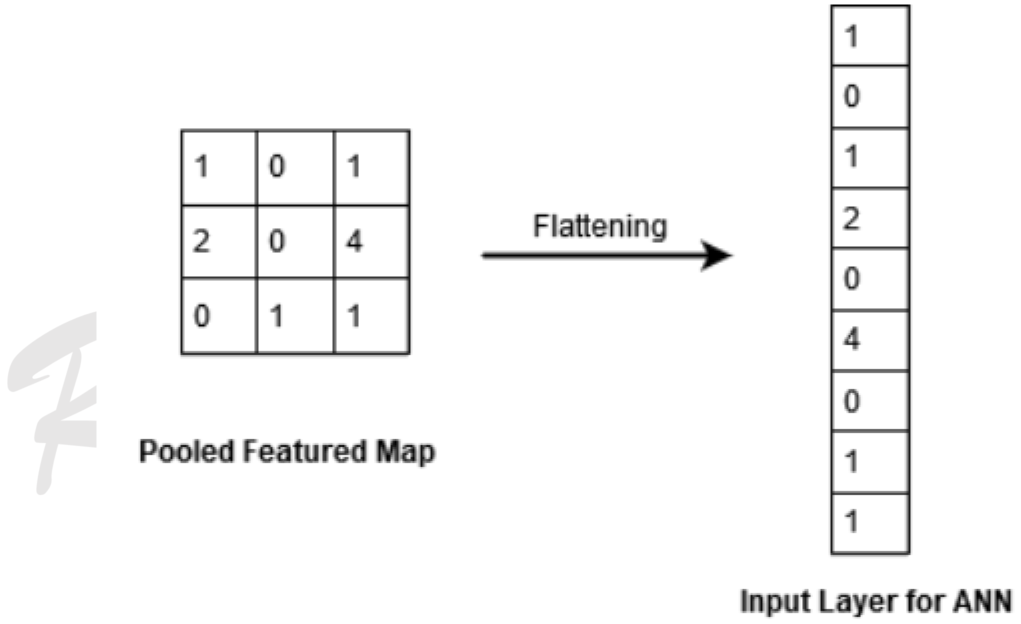
MAXPOOLING

- If same image be rotated or strectched or shrinked, Neural Network should be able to recognize it.
- preserving the feature
- reducing size
- reducing parameter so preventing over-fitting



FLATTENING

- converting the data into a 1-dimensional array for inputting it to the next layer



FULLCONNECTION

- in Artificial NN its hidden layer
- in ANN we have one output - regression problem
- in CNN we have multiple output - classification problem

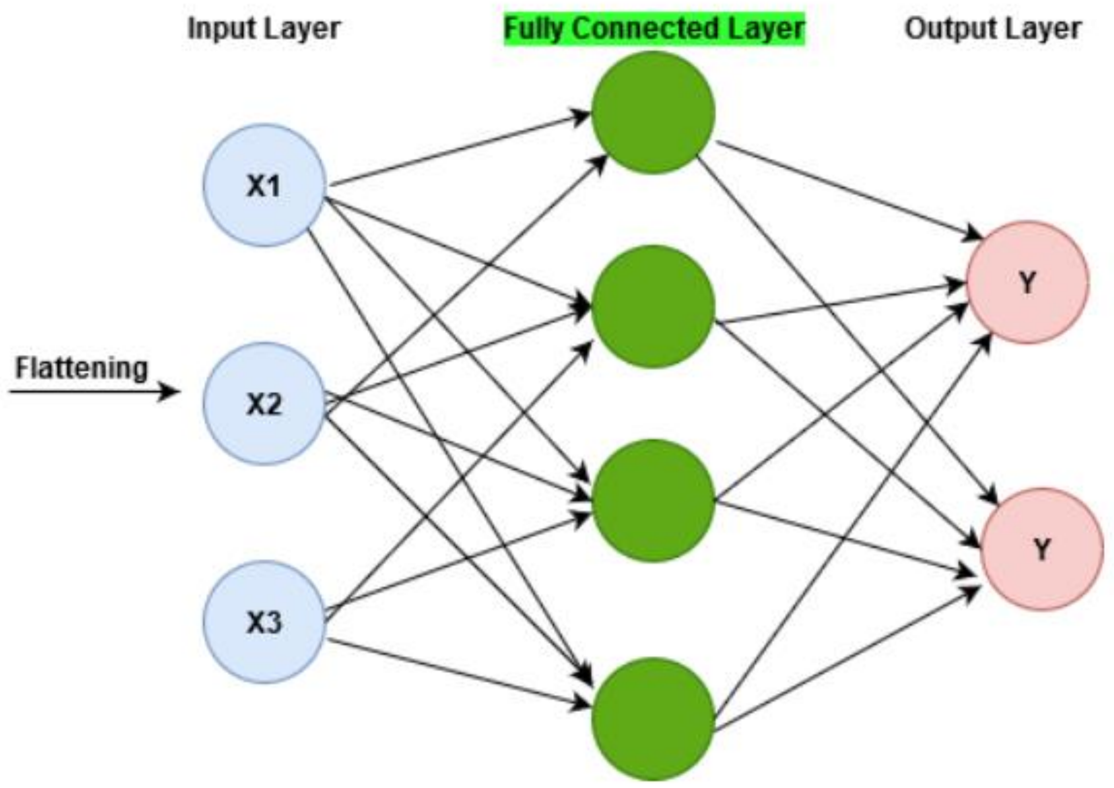


Image Augmentation

- Function - prevent overfitting by providing more dataset to train model
- It creates many batches
- In each batch - it create images from existing images by random transformation(flipping, rotating, shifting, etc) after selecting images randomly

- Keras

from keras.preprocessing.image import ImageDataGenerator

Keras

Building Deep Learning Framework with Keras 2.0

- Keras programming framework to implement Deep Learning
- Breakthrough - image recognition, language translation and speech translation
- Keras - behind uses either tensorflow or theano for all the calculations
- Keras - built in image recognition

Theano vs Tensorflow

- Model built on Keras can work on both, config change allow to switch easily
- Both Support GPU accelation, TF can be run on multiple systems
- TF works with google cloud ML platform

Tensorflow

- Why dont use only TF?
- low level, more control , need to write more code
- Keras - high level, fast experimentation, very less coding

Creating Env

1> Create virtual env

#conda create -n tensorflow pip python=3.5

2> activate env

#activate tensorflow

#conda info --envs

3> Install tensorflow

#conda install -c conda-forge tensorflow

#python -m pip install --upgrade pip

#pip install setuptools

Creating Env

4> Install keras

#pip install keras

5> Install other package

#pip install matplotlib

#pip install sklearn

#pip install pydot

6> Install spyder separately so that you can launch it without activating your virtual env

#conda install spyder