Week 3

Machine Learning Specific

Week 1	Prerequisite Learning
Week 2	Programming Fundamentals (Python) + Required Installation
Week 3	ML Specific
Week 4	Coding
Week 5	Git Hub Deployment
Week 6	Extension + Summarization

An Introduction to Machine Learning

The field of study known as machine learning is concerned with the question of how to construct computer programs that automatically improve with experience.

Examples

- A robot driving learning problem
- Handwriting recognition learning problem

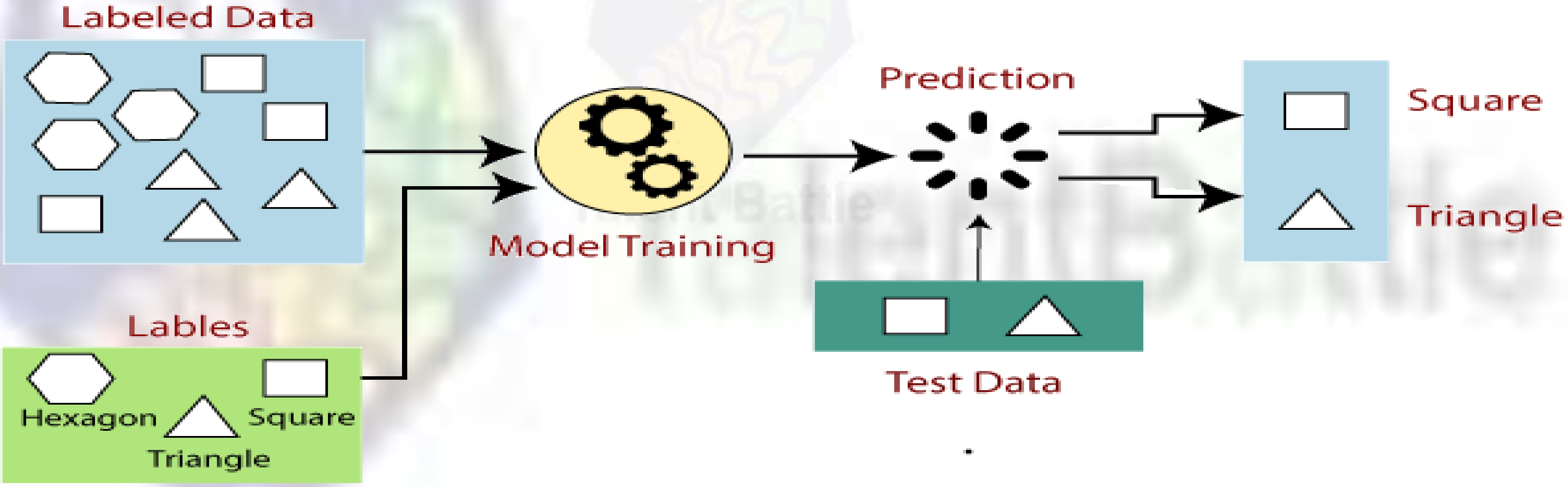
A computer program which learns from experience is called a machine learning program or simply a learning program .

Classification of Machine Learning

1. **Supervised Learning**
2. **Unsupervised Learning**
3. **Reinforcement Learning**
4. **Semi-Supervised Learning**

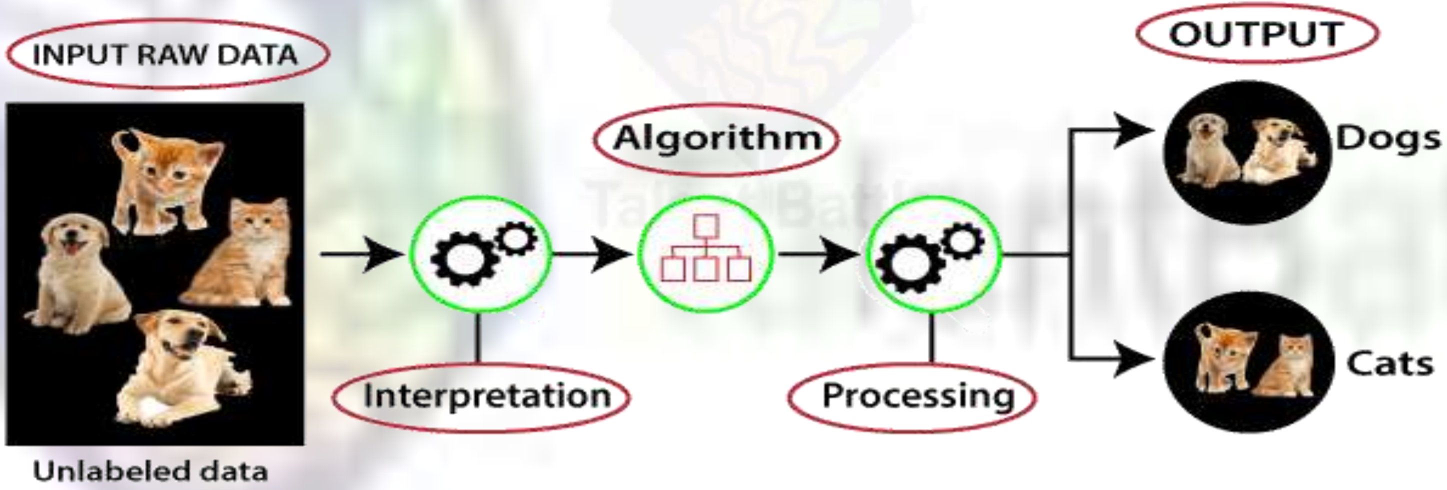
Supervised learning:

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. The given data is labeled .
Both classification and regression problems are supervised learning problems .


Labeled Data
Lables
Hexagon
Square
Triangle
Model Training
Prediction
Test Data
Square
Triangle

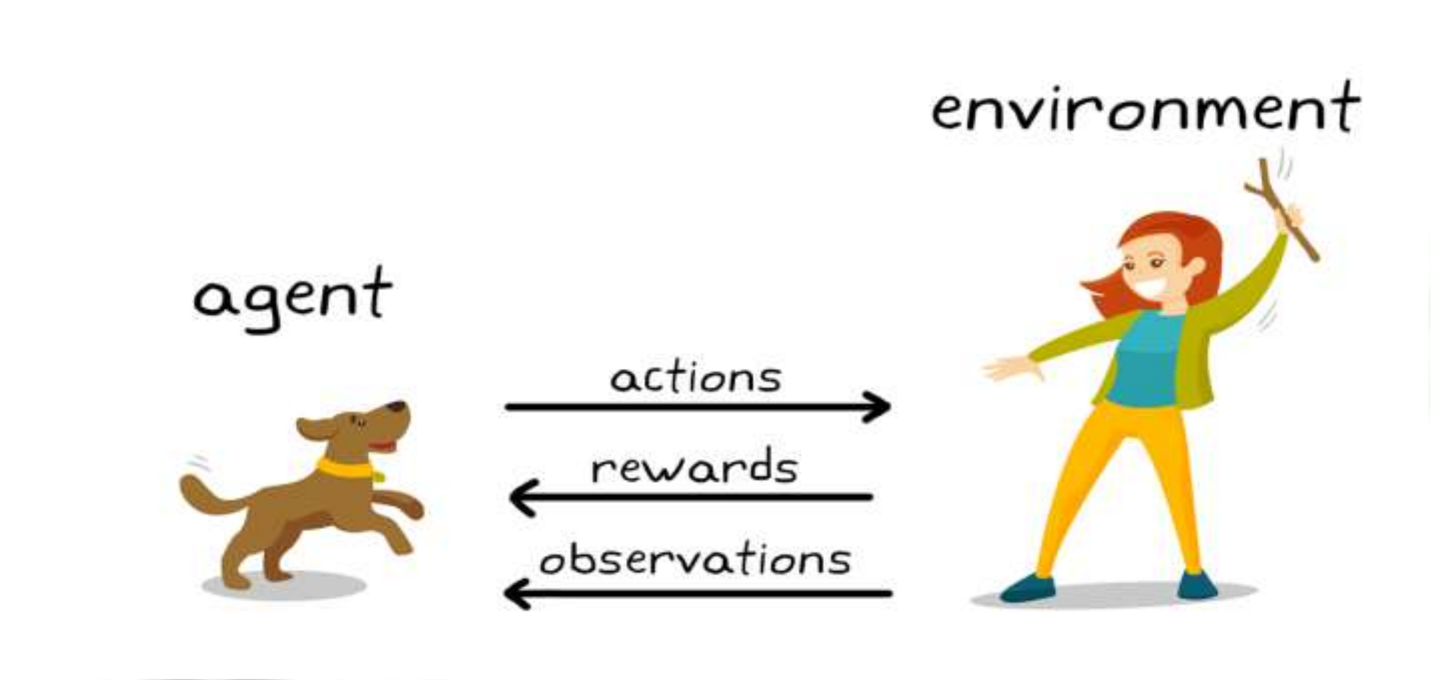
Unsupervised learning:

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.
In unsupervised learning algorithms, classification or categorization is not included in the observations.


INPUT RAW DATA
Unlabeled data
Interpretation
Algorithm
Processing
OUTPUT
Dogs
Cats

Reinforcement learning:
Reinforcement learning is the problem of getting an agent to act in the world so as to maximize its rewards.

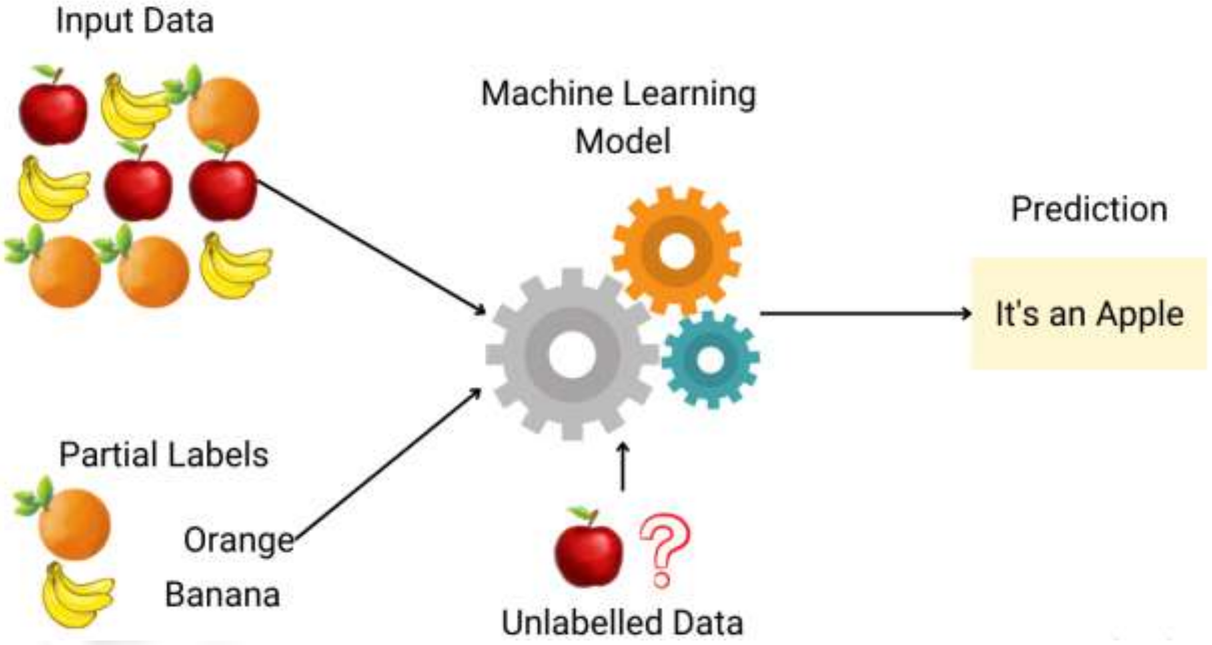
A learner is not told what actions to take as in most forms of machine learning but instead must discover which actions yield the most reward by trying them.


environment
agent
actions
rewards
observations

Semi-supervised learning:

Semi-supervised learning is an approach to machine learning that combines small labeled data with a large amount of unlabeled data during training.

Semi-supervised learning falls between unsupervised learning and supervised learning.


Input Data
Machine Learning Model
Prediction
It's an Apple
Partial Labels
Orange
Banana
Unlabelled Data

Machine Learning Model Development Steps

1. Collecting Data
2. Preparing the Data
3. Choosing a Model
4. Training the Model
5. Evaluating the Model
6. Parameter Tuning
7. Making Predictions

In this project, we are going to use the MNIST dataset for the implementation of a handwritten digit recognition app.

To implement this we will use a special type of deep neural network called *Convolutional Neural Networks*.

In the end, we will also build a Graphical user interface(GUI) where you can directly draw the digit and recognize it straight away.

What is Handwritten Digit Recognition?

Handwritten digit recognition is the process to provide the ability to machines to recognize human handwritten digits.

It is not an easy task for the machine because handwritten digits are not perfect, vary from person-to-person, and can be made with many different flavors.

Prerequisites

Basic knowledge of deep learning with Keras library, the Tkinter library for GUI building, and Python programming are required to run this amazing project.

The MNIST dataset

Among thousands of datasets available in the market, MNIST is the most popular dataset for enthusiasts of machine learning and deep learning. Above 60,000 plus training images of handwritten digits from zero to nine and more than 10,000 images for testing are present in the MNIST dataset.

So, 10 different classes are in the MNIST dataset. The images of handwritten digits are shown as a matrix of 28×28 where every cell consists of a grayscale pixel value.

Steps to build Handwritten Digit Recognition System

1. Import libraries and dataset

At the project beginning, we import all the needed modules for training our model. We can easily import the dataset and start working on that because the Keras library already contains many datasets and MNIST is one of them. We call mnist.load_data() function to get training data with its labels and also the testing data with its labels.

2. The Data Preprocessing

Model cannot take the image data directly so we need to perform some basic operations and process the data to make it ready for our neural network.

The dimension of the training data is (60000*28*28).

One more dimension is needed for the CNN model so we reshape the matrix to shape (60000*28*28*1).

3. Create the model

A convolutional layer and pooling layers are the two wheels of a CNN model. The reason behind the success of CNN for image classification problems is its feasibility with grid structured data. We will use the Adadelta optimizer for the model compilation.

Adadelta optimization is **a stochastic gradient descent method that is based on adaptive learning rate per dimension** to address two drawbacks: The continual decay of learning rates throughout training. The need for a manually selected global learning rate.

4. Train the model

To start the training of the model we can simply call the model.fit() function of Keras. It takes the training data, validation data, epochs, and batch size as the parameter.

5. Evaluate the model

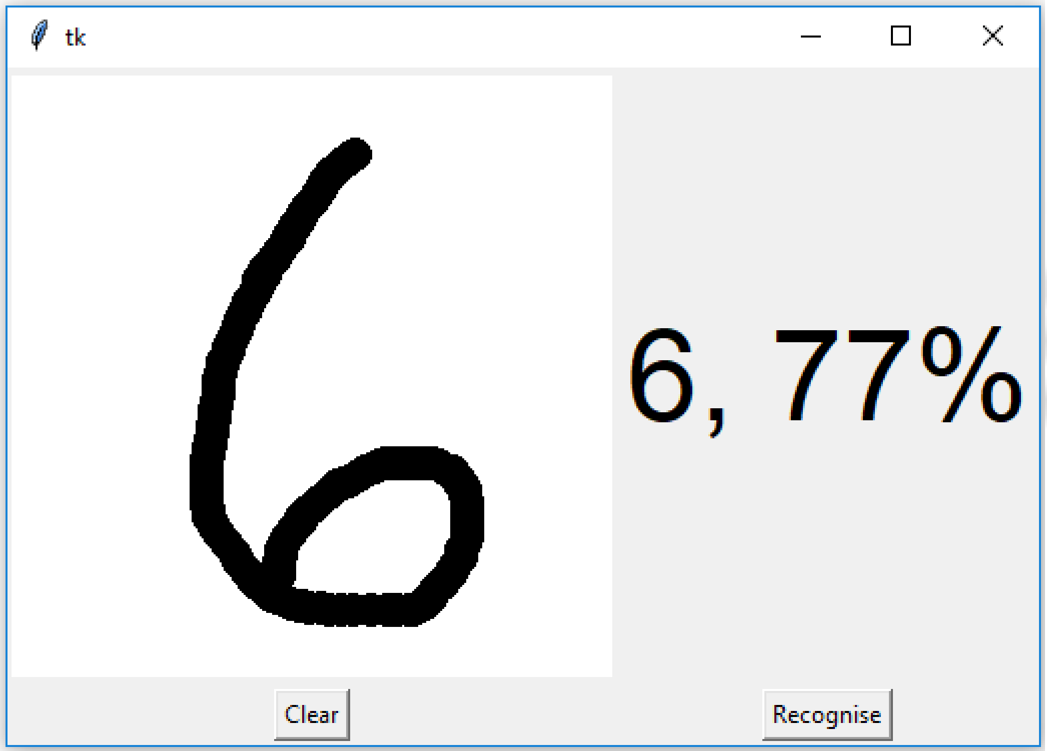
To evaluate how accurate our model works, we have around 10,000 images in our dataset.

In the training of the data model, we do not include the testing data that’s why it is new data for our model.

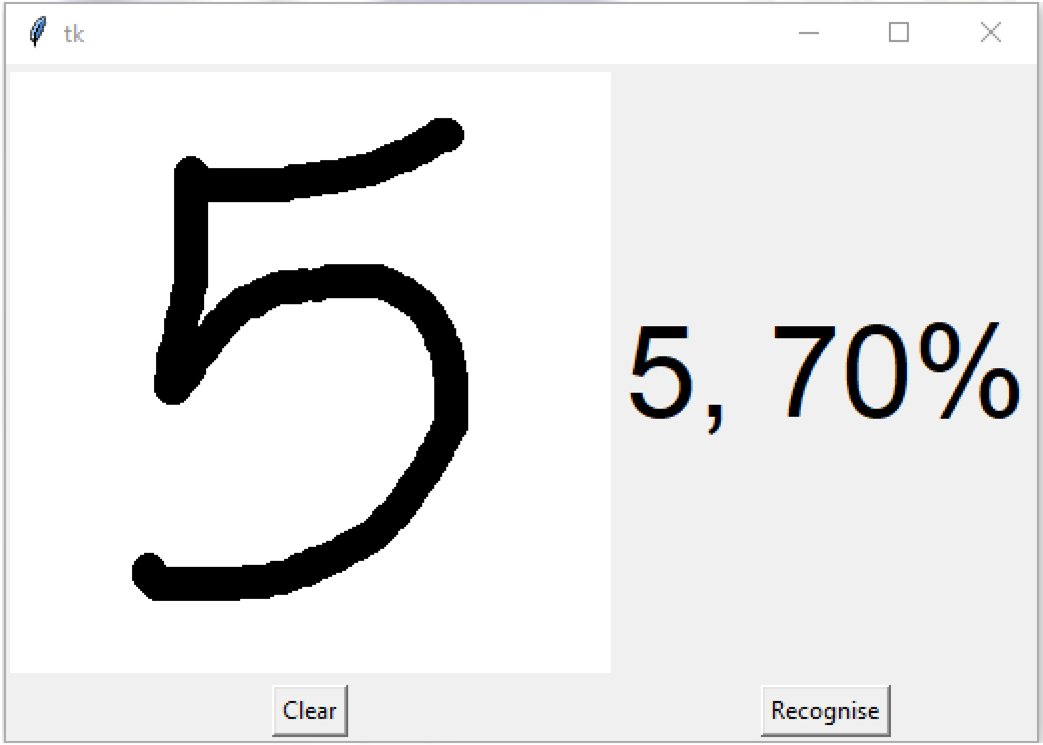
6. Create GUI to predict digits

To build an interactive window we have created a new file in GUI. In this file, you can draw digits on canvas, and by clicking a button, you can identify the digit. The Tkinter library is the part of Python standard library. Our predict_digit() method takes the picture as input and then activates the trained model to predict the digit.

After that to build the GUI for our app we have created the App class. In GUI canvas you can draw a digit by capturing the mouse event and with a button click, we hit the predict_digit() function and show the results.



```
6, 77%
```



```
5, 70%
```

Imp Terms:

1. Deep Learning
2. Artificial Neural Network
3. CNN

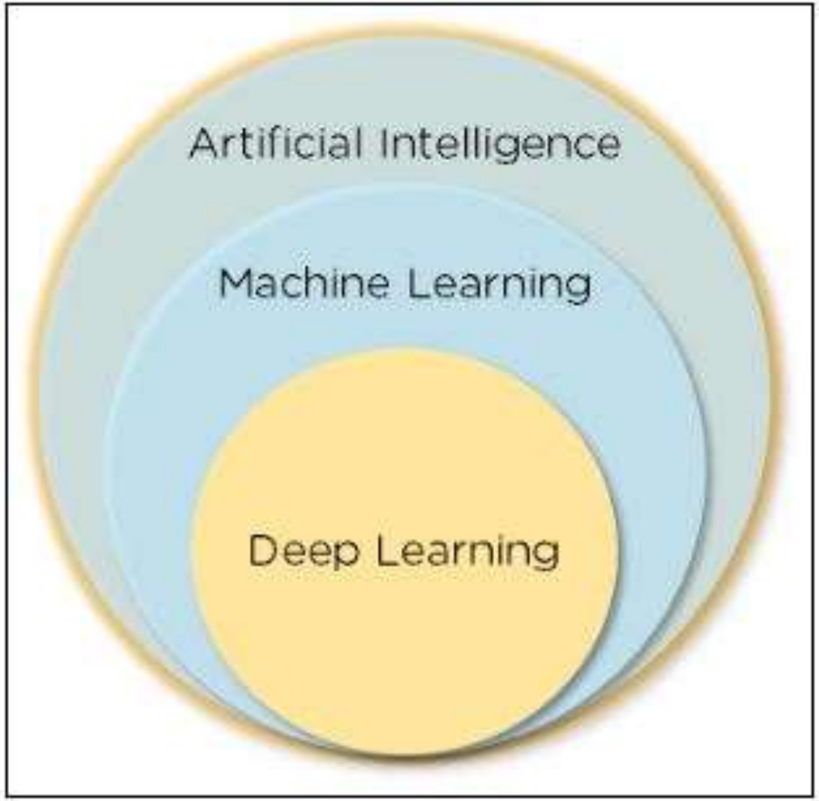
Deep learning is a branch of artificial intelligence concerned with solving highly complex problems by emulating the working of the human brain.

In deep learning, we use neural networks which use multiple operators placed in nodes to help break down the problem into smaller parts, which are each solved individually. But neural networks can be really hard to implement. This problem is taken care of by Keras, a deep learning framework.

What Is Keras?

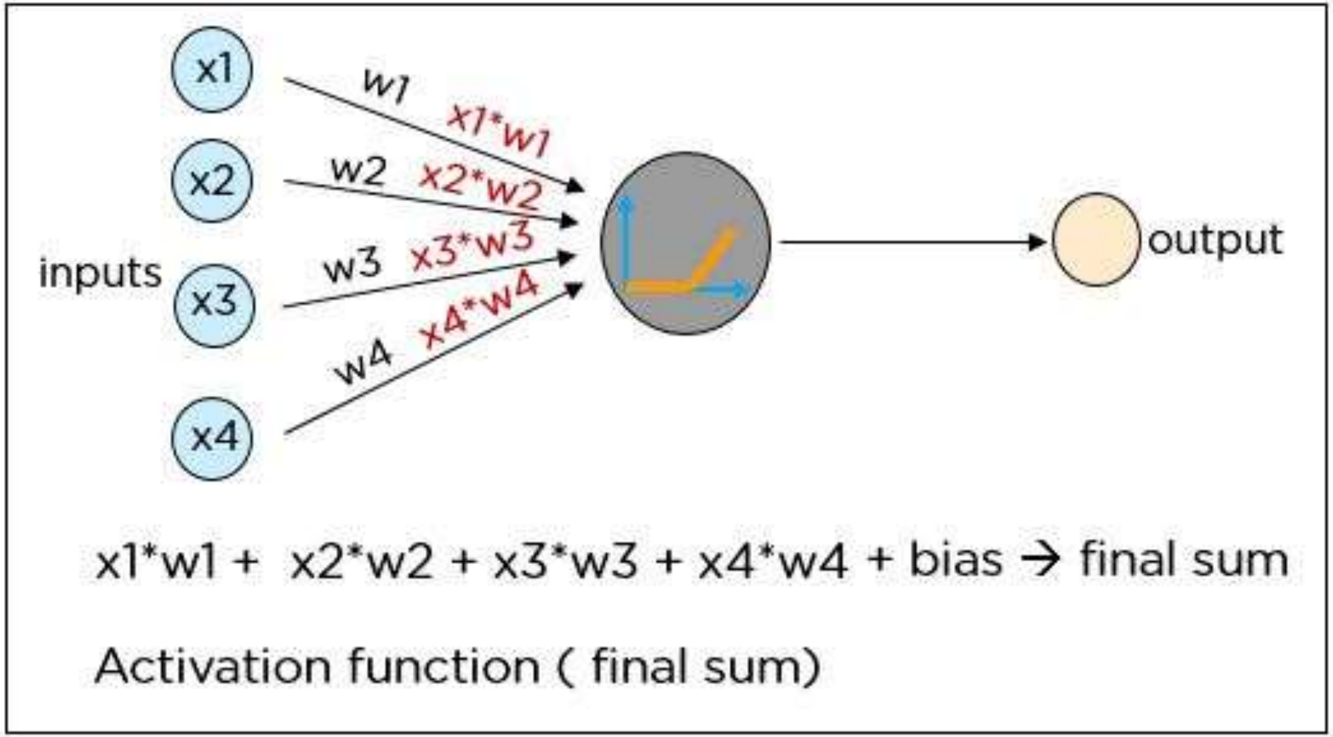
Keras is a high-level, deep learning API developed by Google for implementing neural networks. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation.

Keras is relatively easy to learn and work with because it provides a python frontend with a high level of abstraction while having the option of multiple back-ends for computation purposes.


Artificial Intelligence
Machine Learning
Deep Learning

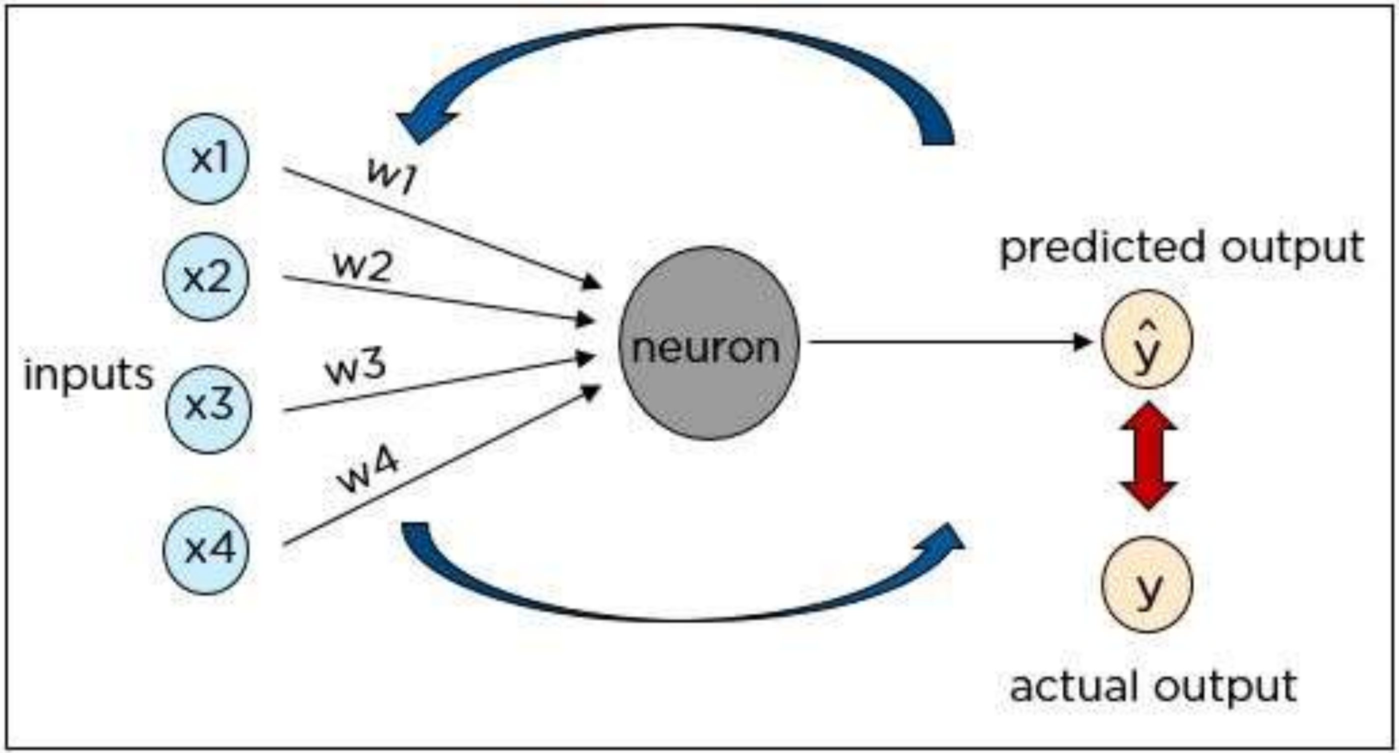
What are Neural Networks?

A neural network is a system modeled on the human brain, consisting of an input layer, multiple hidden layers, and an output layer. Data is fed as input to the neurons. The information is transferred to the next layer using appropriate weights and biases. The output is the final value predicted by the artificial neuron.

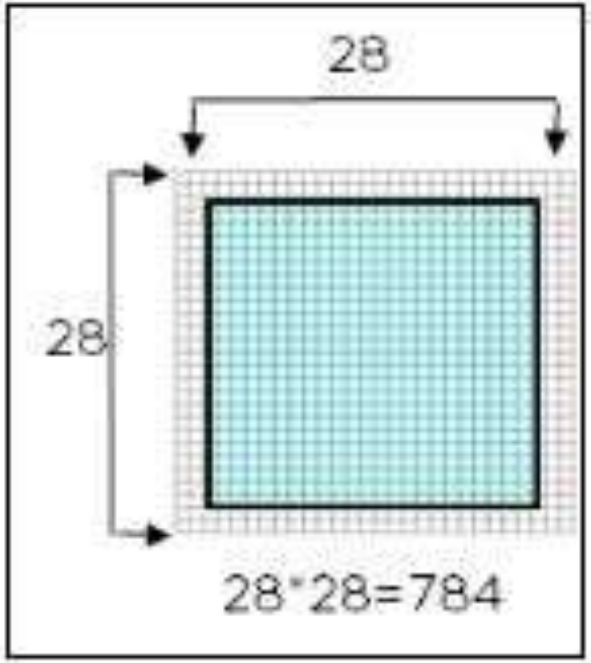


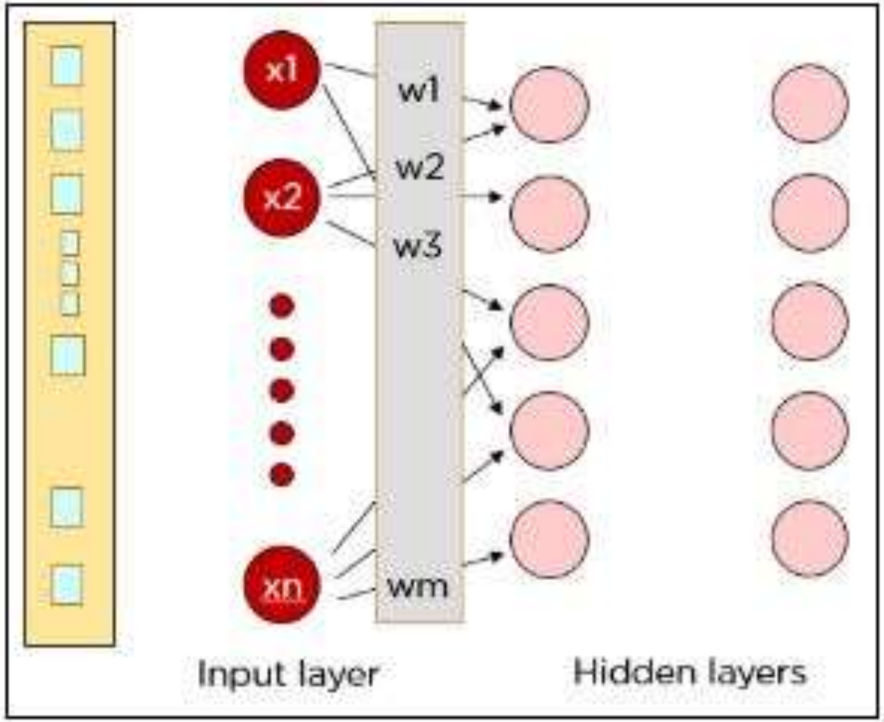
x1*w1 + x2*w2 + x3*w3 + x4*w4 + bias → final sum

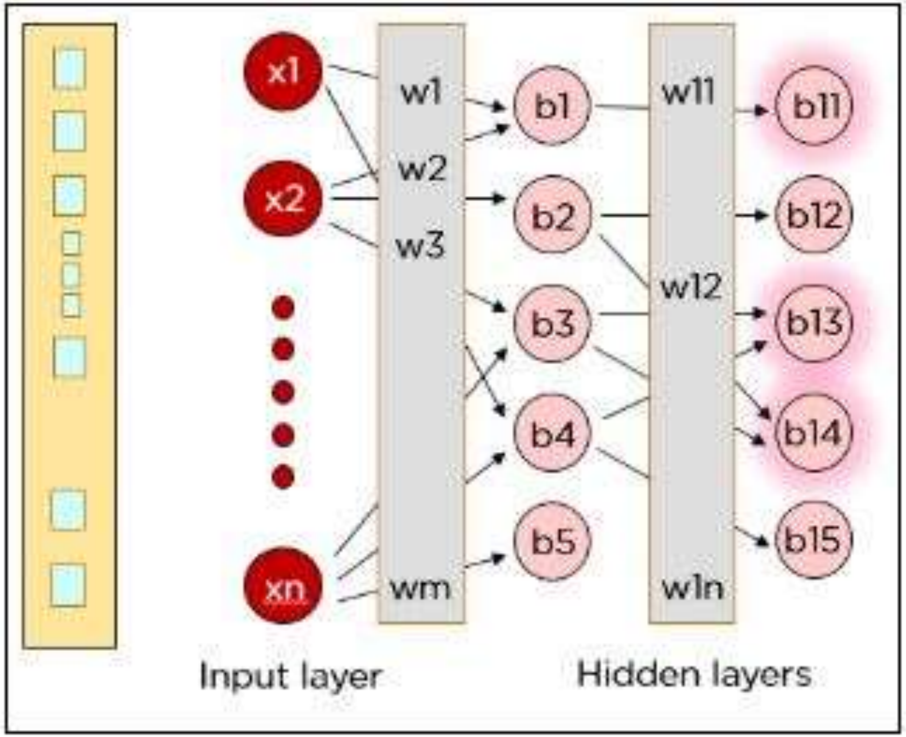
Activation function (final sum)

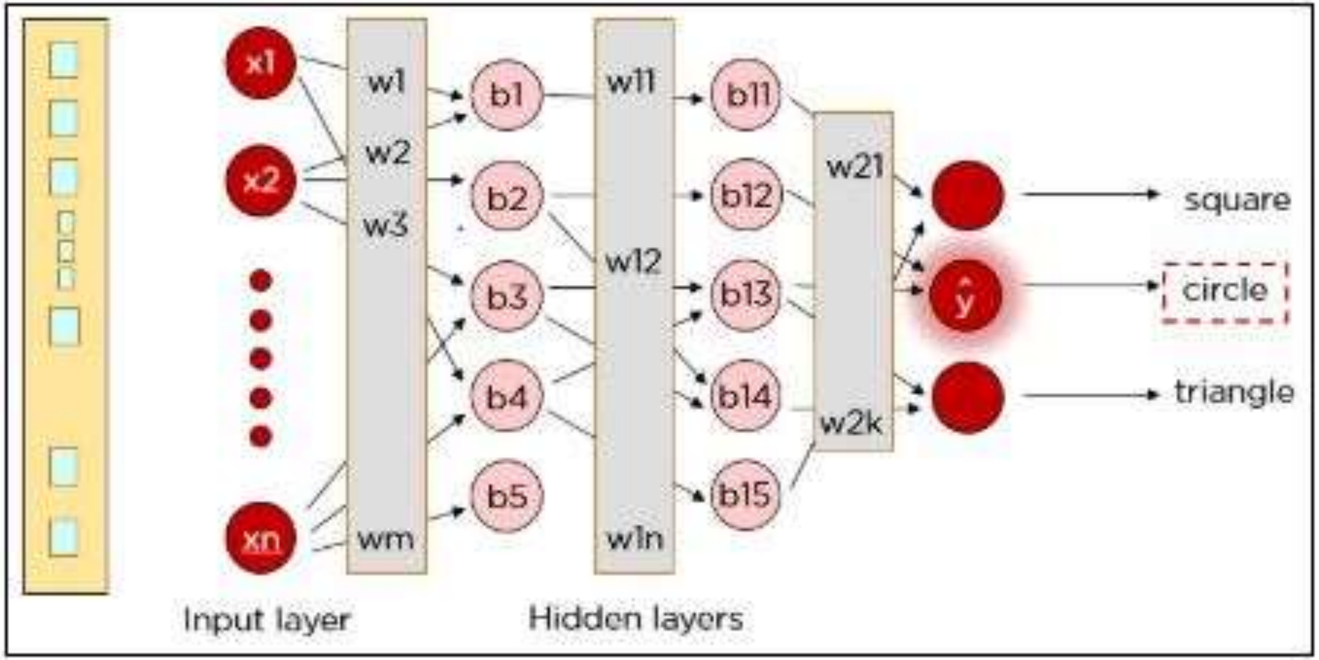

x1
x2
x3
x4
inputs
w1
w2
w3
w4
neuron
predicted output
ŷ
y
actual output

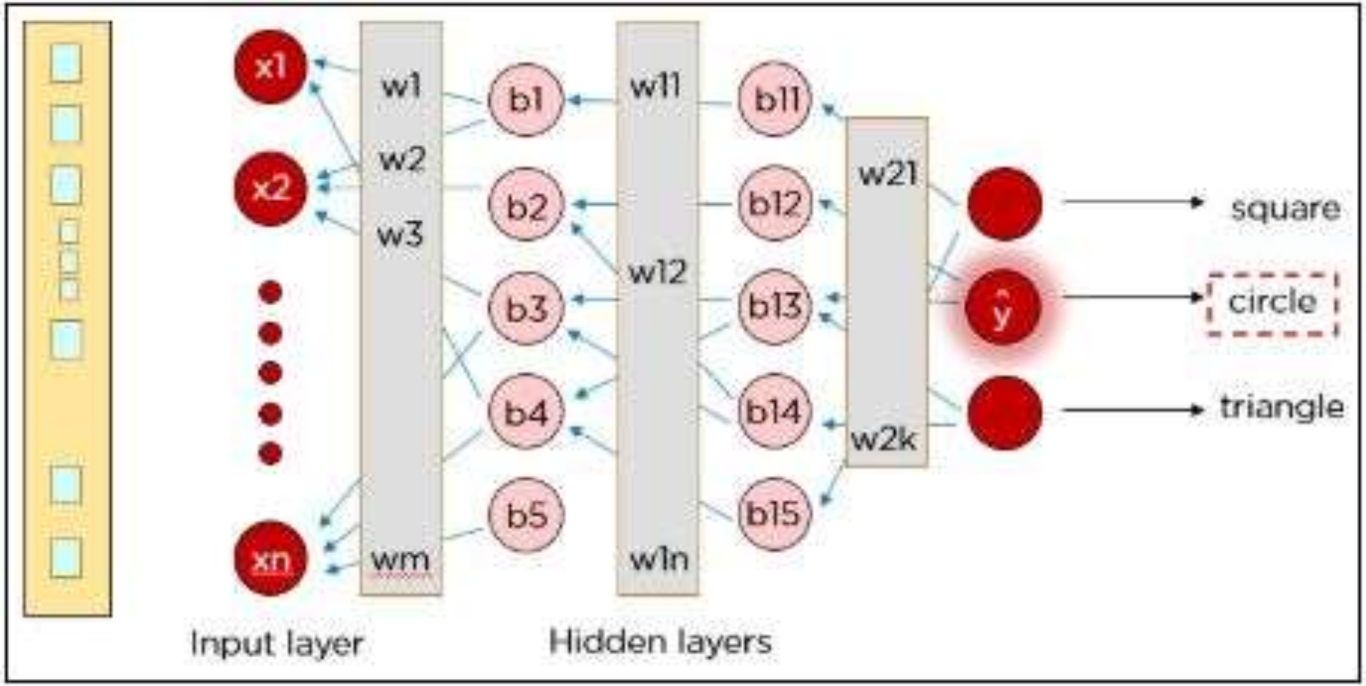
How Do Neural Networks Work?

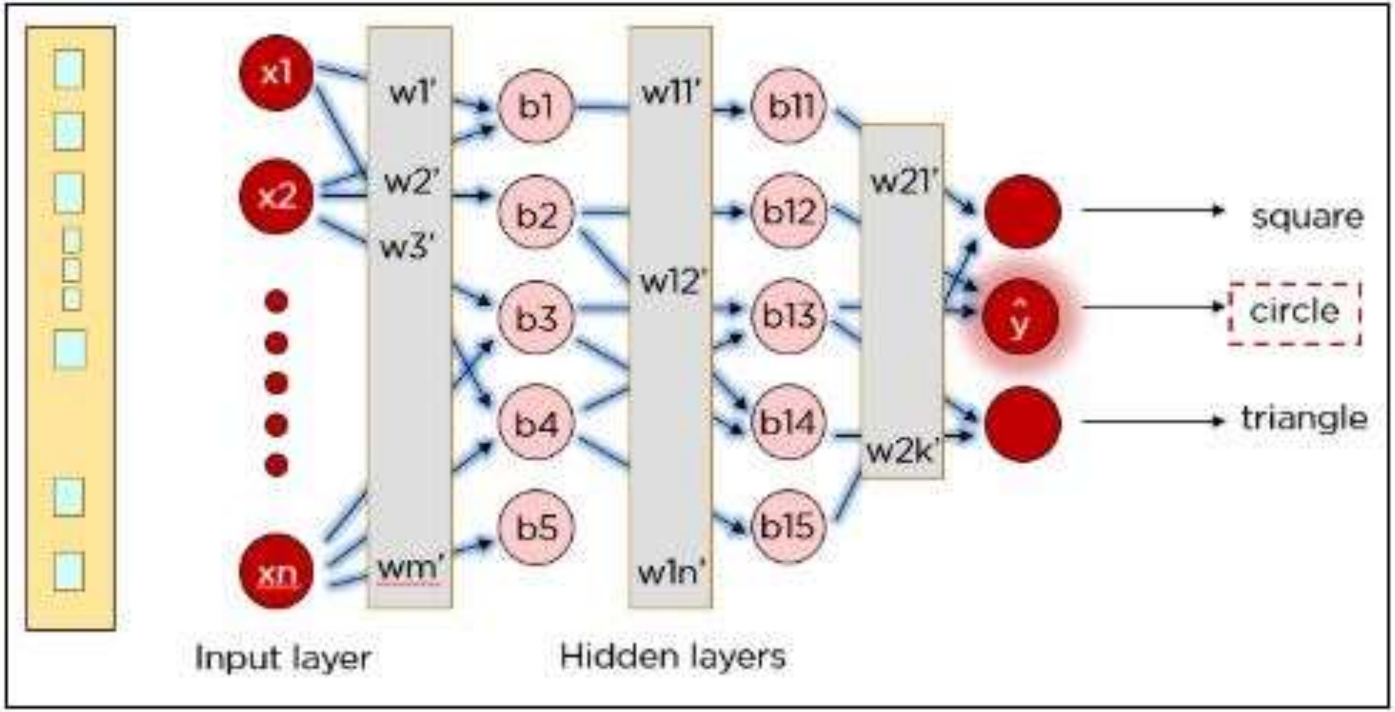


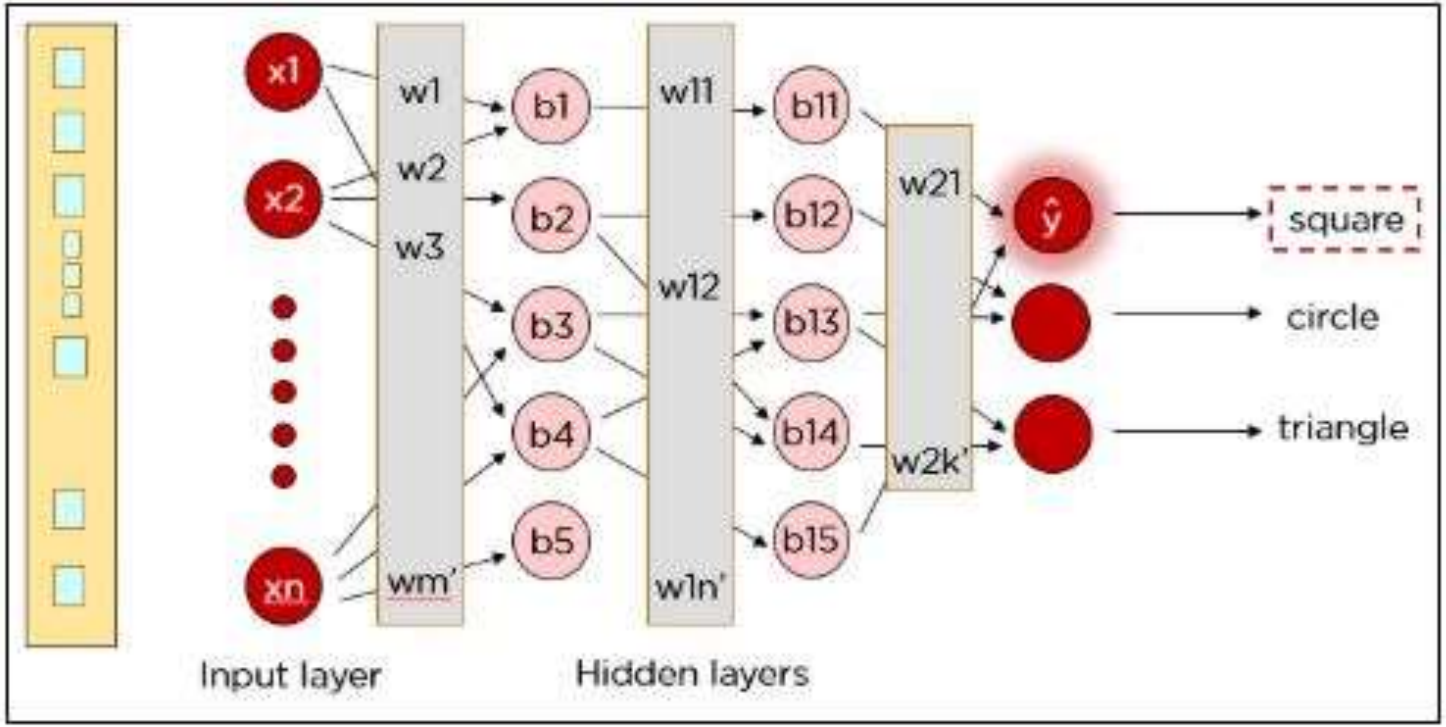

x1
x2
xn
w1
w2
w3
wm
Input layer
Hidden layers


x1
x2
xn
w1
w2
w3
wm
b1
b2
b3
b4
b5
w11
w12
w1n
b11
b12
b13
b14
b15
Input layer
Hidden layers


x1
x2
xn
w1
w2
w3
wm
b1
b2
b3
b4
b5
w11
w12
w1n
b11
b12
b13
b14
b15
w21
w2k
ŷ
square
circle
triangle
Input layer
Hidden layers


x1
x2
xn
w1
w2
w3
wm
b1
b2
b3
b4
b5
w11
w12
w1n
b11
b12
b13
b14
b15
w21
w2k
ŷ
square
circle
triangle
Input layer
Hidden layers

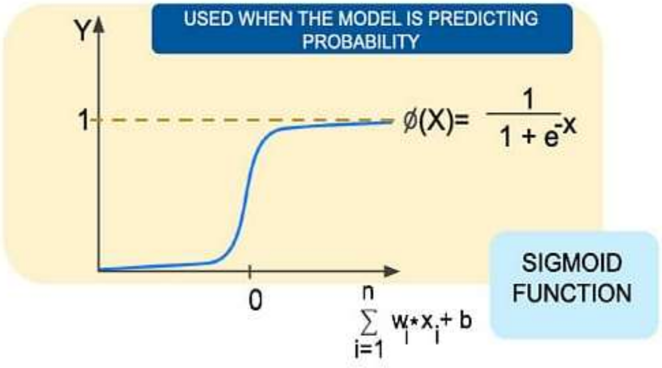

x1
x2
xn
w1'
w2'
w3'
wm'
b1
b2
b3
b4
b5
w11'
w12'
w1n'
b11
b12
b13
b14
b15
w21'
w2k'
ŷ
square
circle
triangle
Input layer
Hidden layers


x1
x2
xn
w1
w2
w3
wm'
b1
b2
b3
b4
b5
w11
w12
w1n'
b11
b12
b13
b14
b15
w21
w2k'
ŷ
square
circle
triangle
Input layer
Hidden layers

There are different types of activation functions.

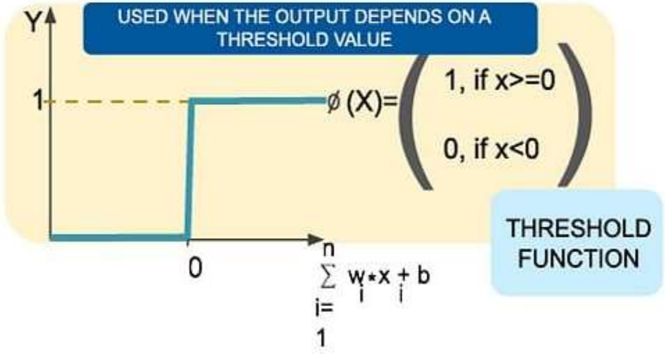
Sigmoid Function

The sigmoid function is used when the model is predicting probability.



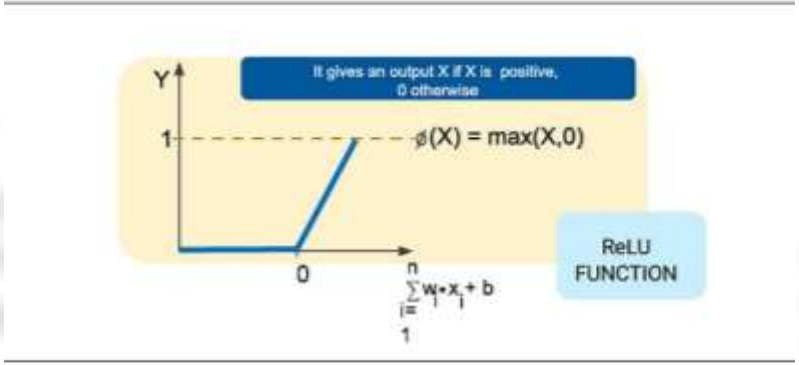
Threshold Function

The threshold function is used when you don’t want to worry about the uncertainty in the middle.


USED WHEN THE OUTPUT DEPENDS ON A THRESHOLD VALUE
Y
1
0
ø (X)=
1, if x>=0
0, if x<0
$\sum_{i=1}^{n} w_i * x_i + b$
THRESHOLD FUNCTION

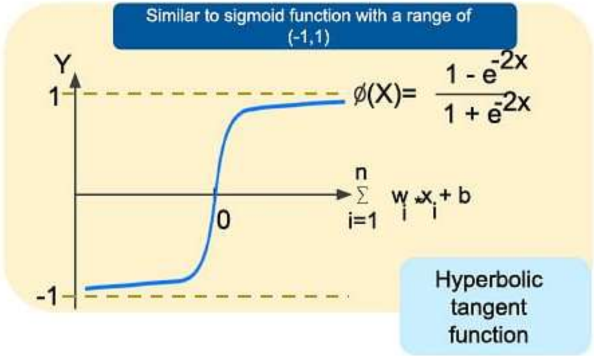
ReLU (rectified linear unit) Function

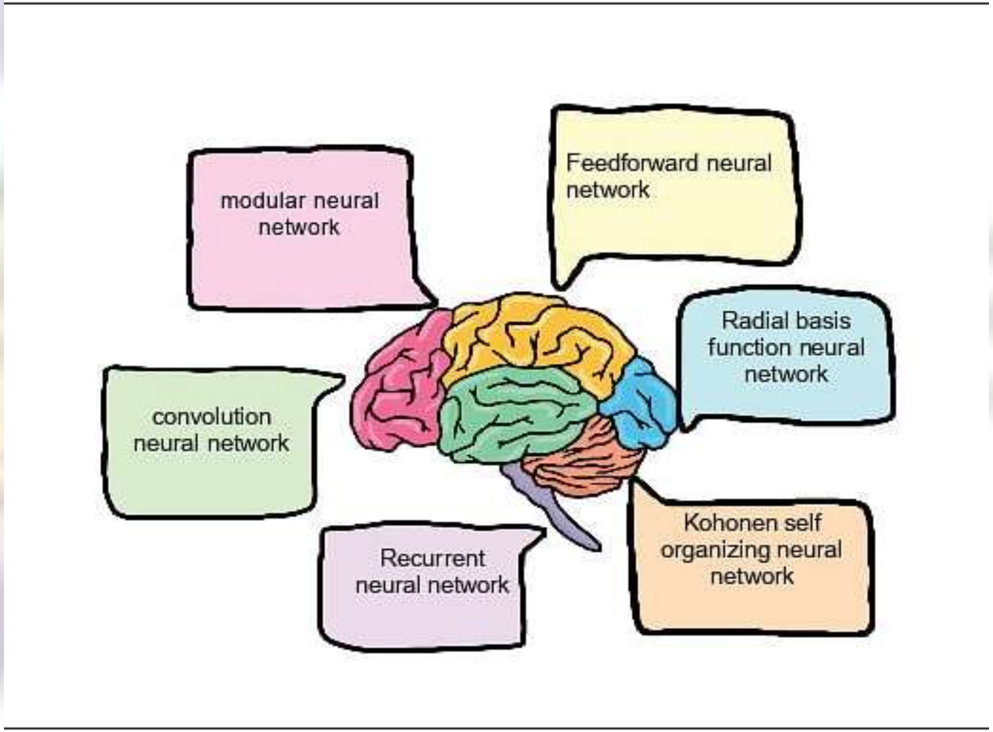
The ReLU (rectified linear unit) function gives the value but says if it’s over 1, then it will just be 1, and if it’s less than 0, it will just be 0. The ReLU function is most commonly used these days.



Hyperbolic Tangent Function

The hyperbolic tangent function is similar to the sigmoid function but has a range of -1 to 1.

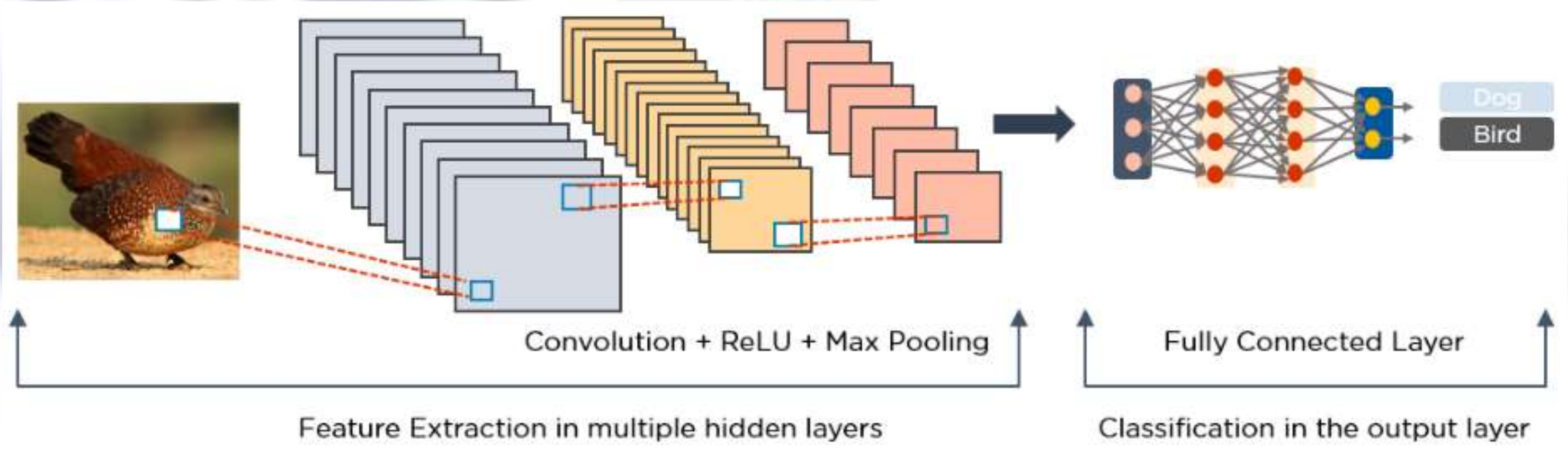


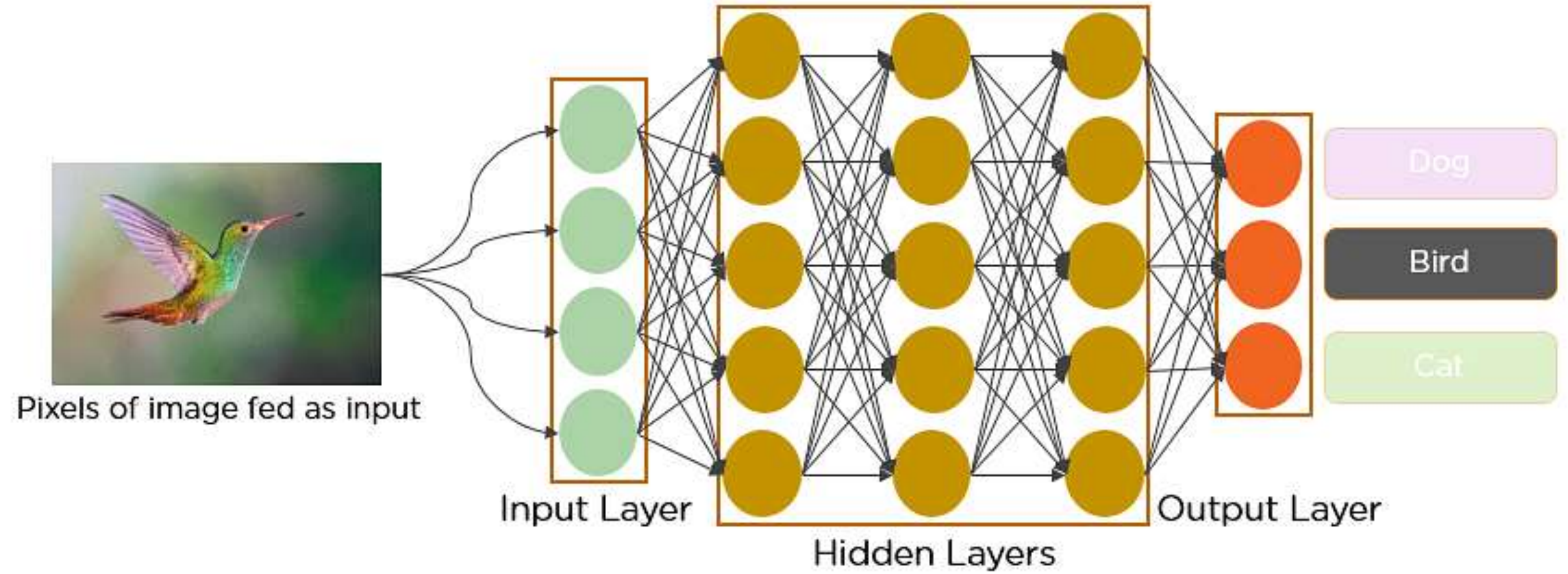

modular neural network
Feedforward neural network
Radial basis function neural network
convolution neural network
Recurrent neural network
Kohonen self organizing neural network

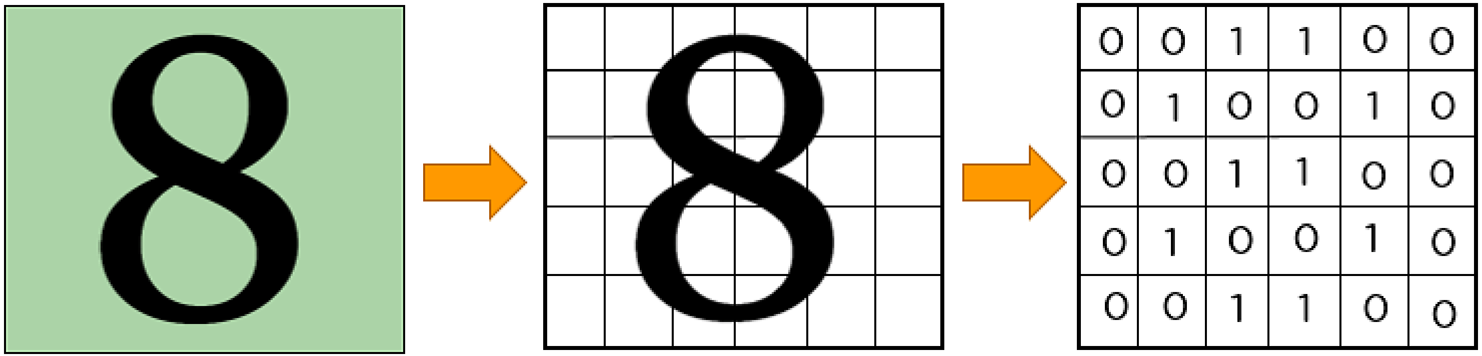
Convolutional Neural Networks (CNNs)

CNN's, also known as ConvNets, consist of multiple layers and are mainly used for image processing and object detection. Yann LeCun developed the first CNN in 1988 when it was called LeNet. It was used for recognizing characters like ZIP codes and digits.

CNN's are widely used to identify satellite images, process medical images, forecast time series, and detect anomalies.


Dog
Bird
Convolution + ReLU + Max Pooling
Fully Connected Layer
Feature Extraction in multiple hidden layers
Classification in the output layer


Pixels of image fed as input
Input Layer
Hidden Layers
Output Layer
Dog
Bird
Cat



Real Image of the digit 8

Represented in the form of an array

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

Digit 8 represented in the form of pixels of 0's and 1's

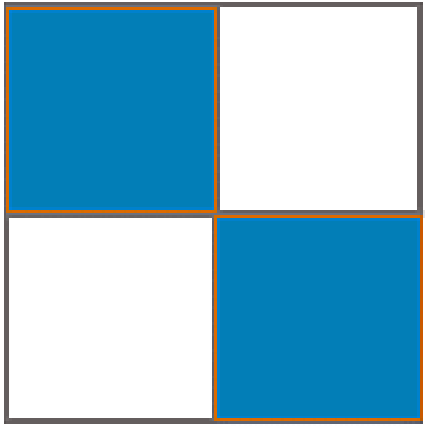


image for the symbol \

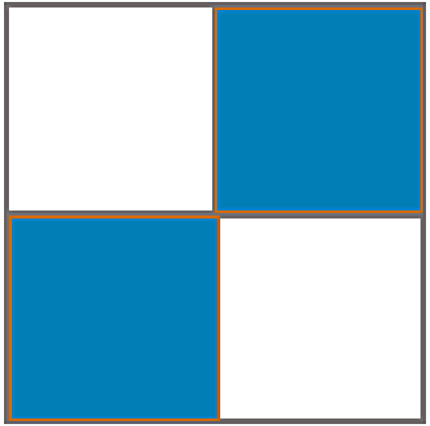
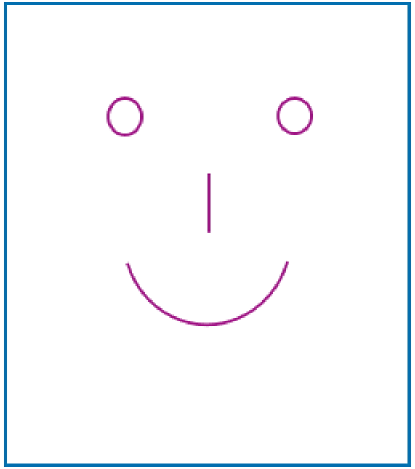
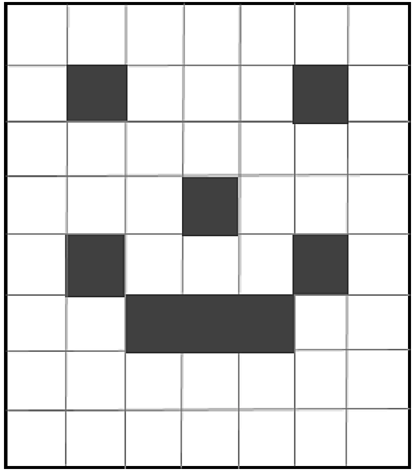


image for the symbol /



Real Image



Represented in the form of black and white pixels

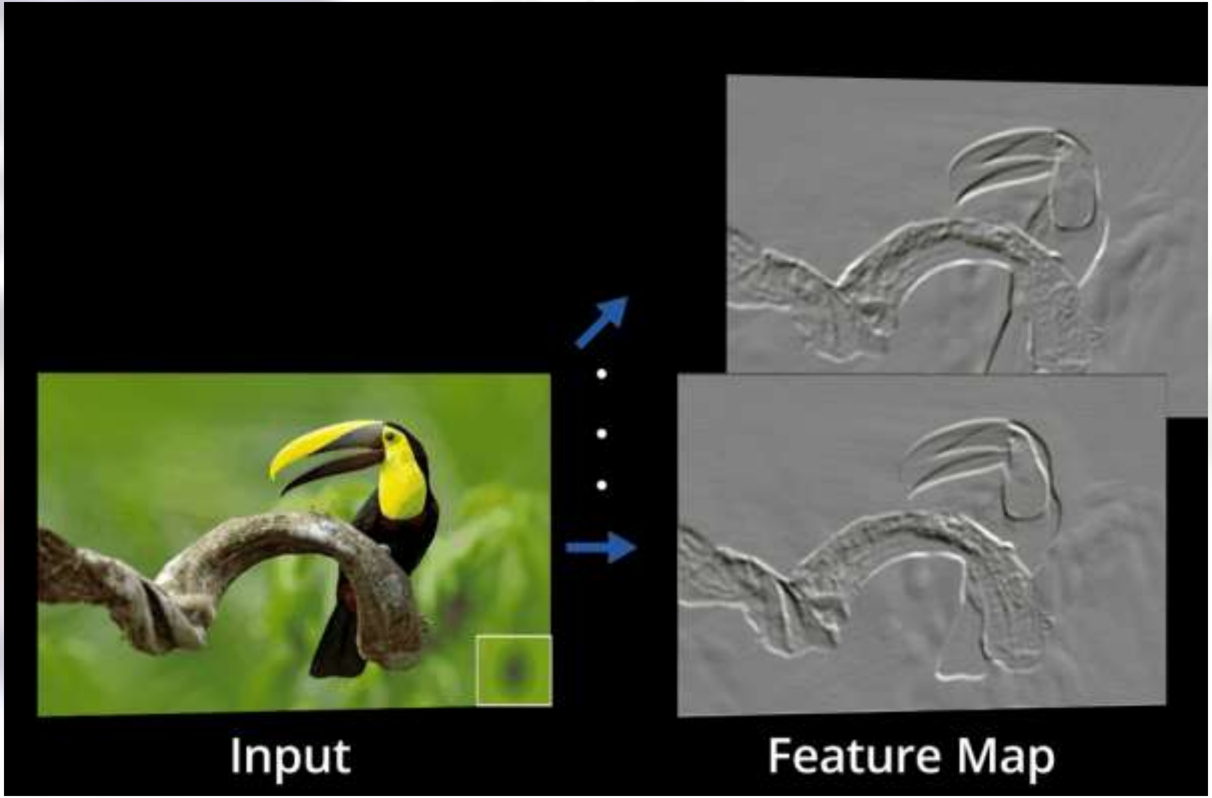
0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

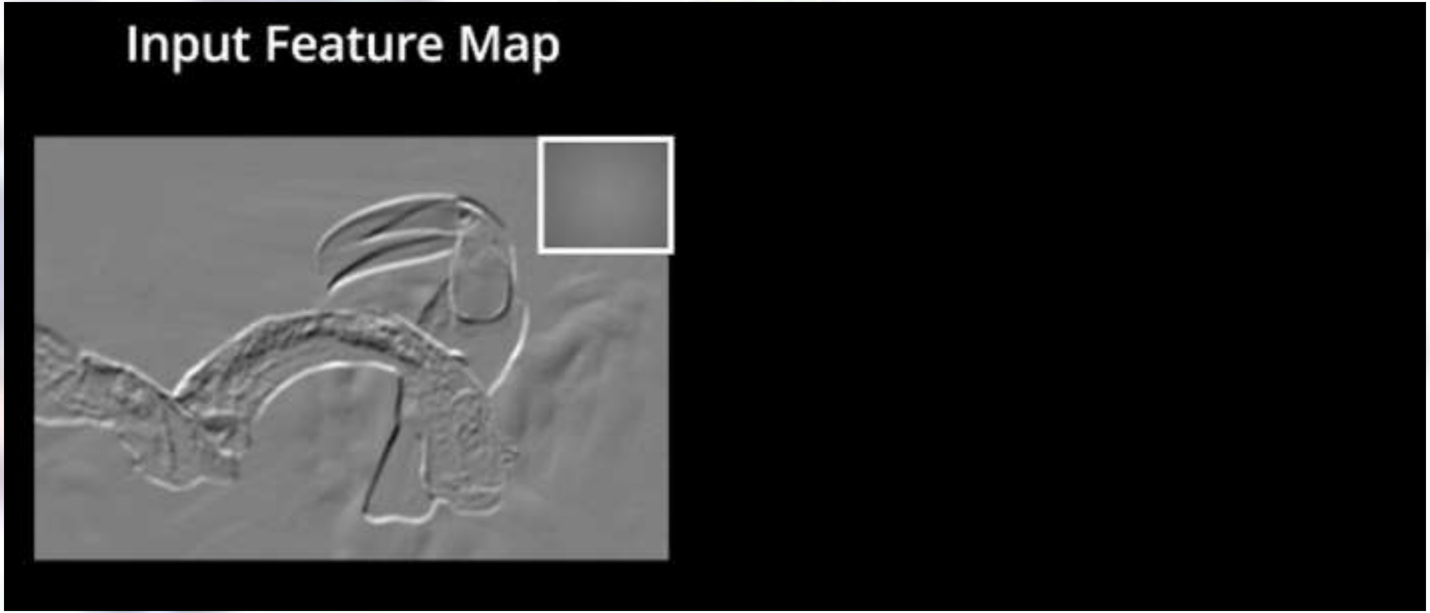
Image represented in the form of a matrix of numbers

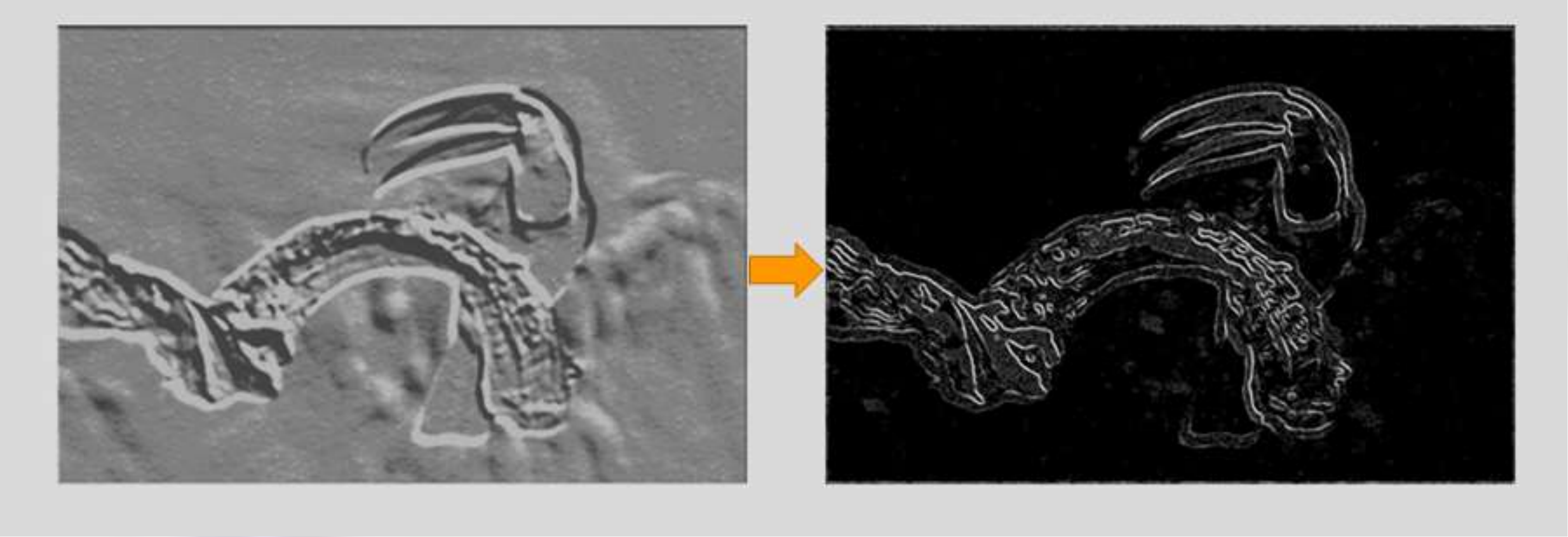
Layers in a Convolutional Neural Network

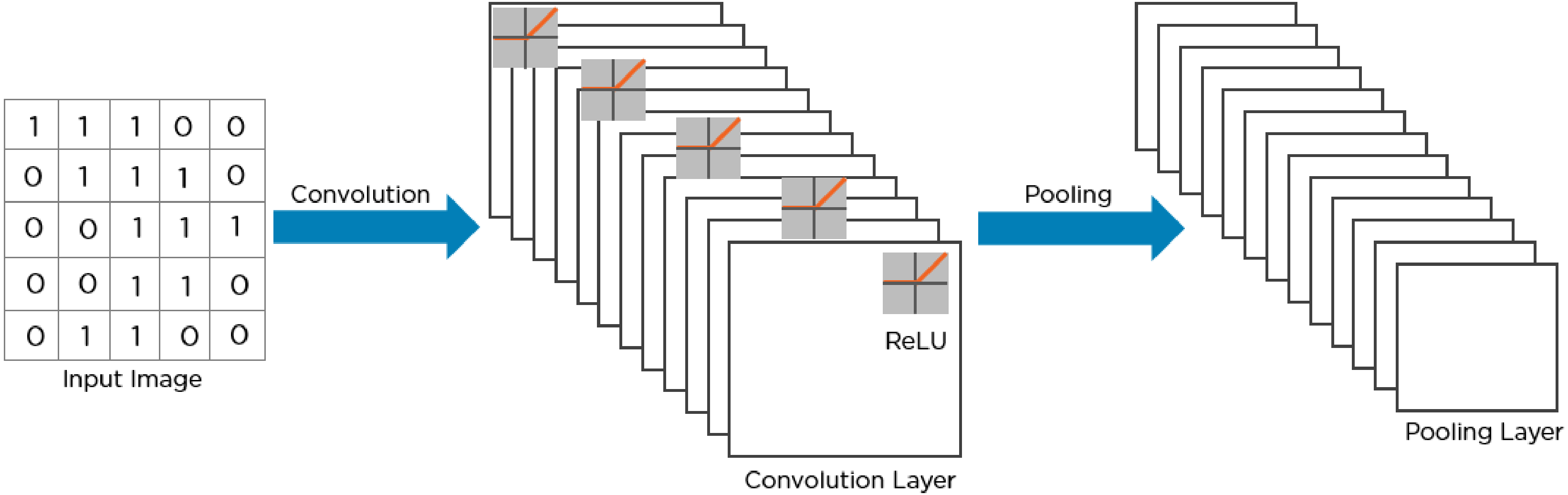
A convolution neural network has multiple hidden layers that help in extracting information from an image. The four important layers in CNN are:

1. Convolution layer
2. ReLU layer
3. Pooling layer
4. Fully connected layer

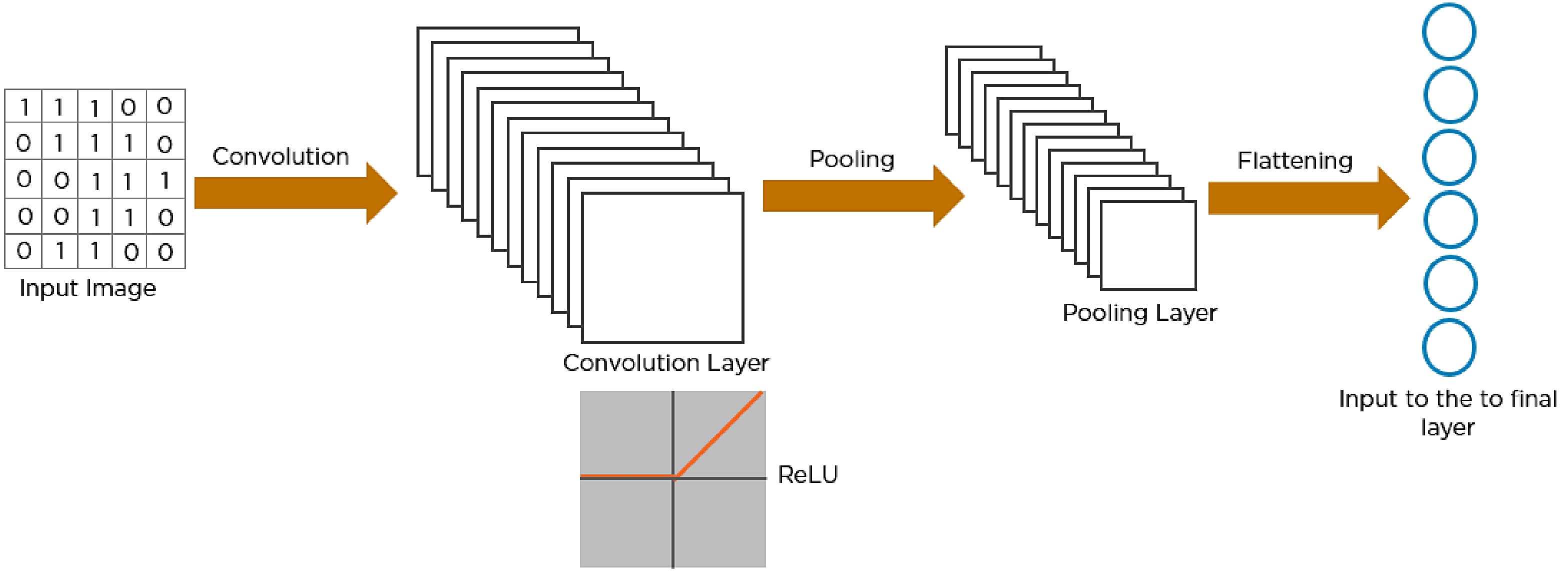

Input
Feature Map


Input Feature Map

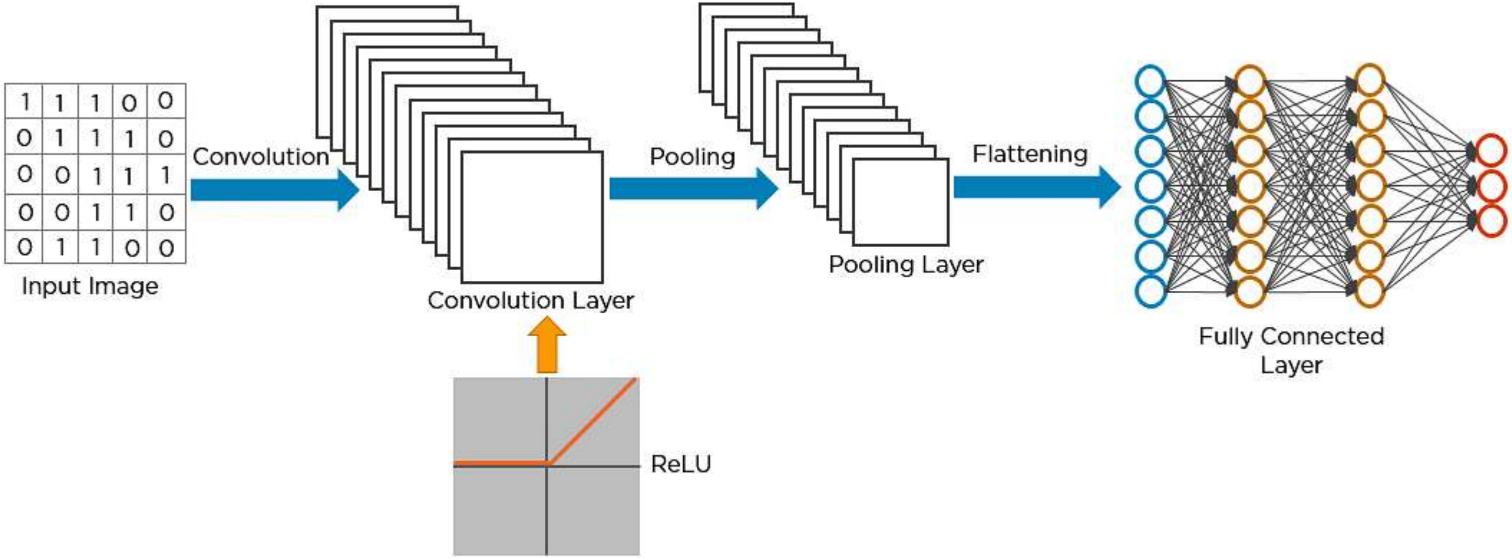




1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

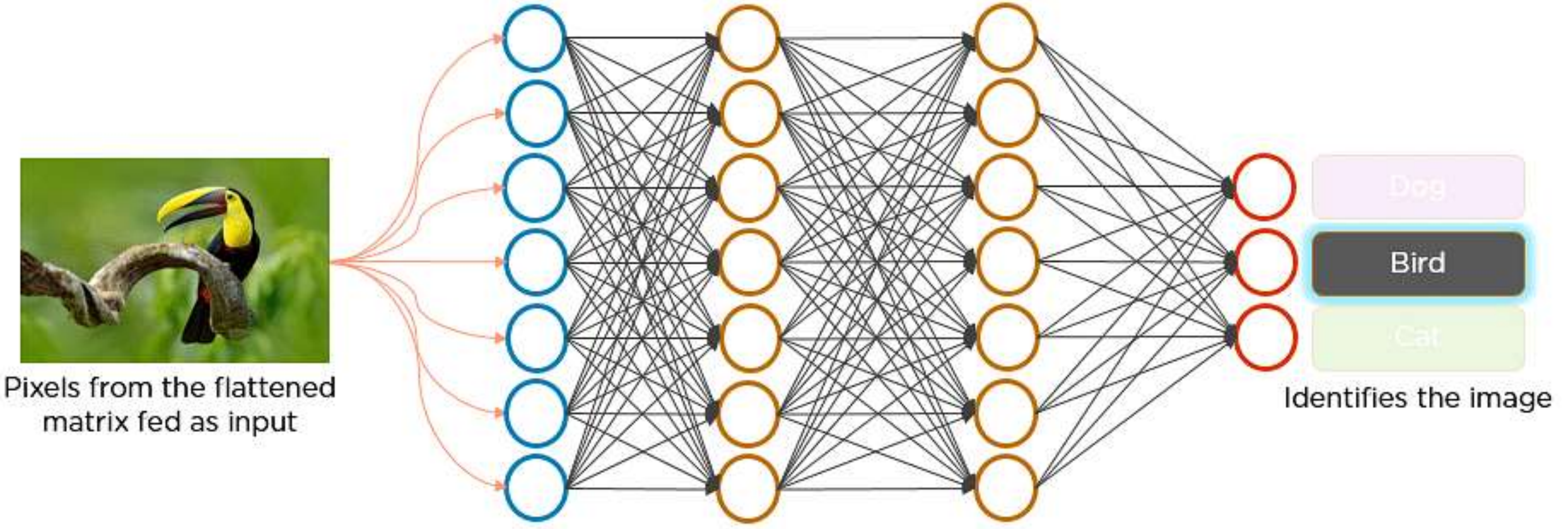


1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0



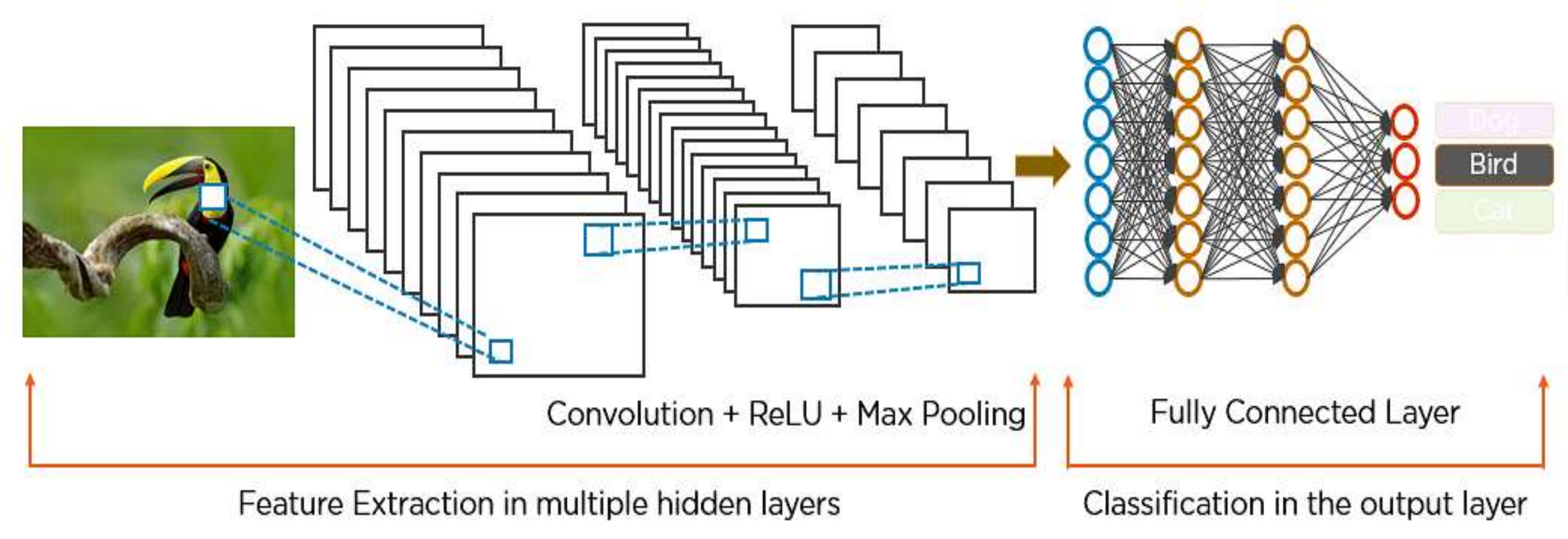
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input Image



Pixels from the flattened matrix fed as input

Identifies the image


Dog
Bird
Cat
Convolution + ReLU + Max Pooling
Fully Connected Layer
Feature Extraction in multiple hidden layers
Classification in the output layer

Thank You !!!