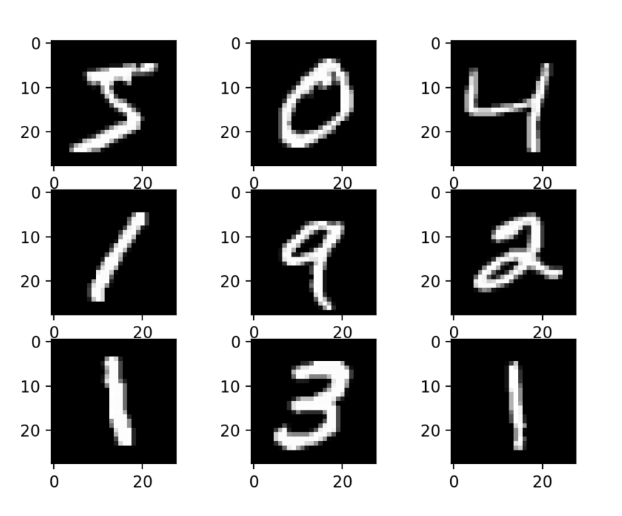
**Hand Written Digits Classification using Deep Neural Network.**

Classifying hand written digits forms to be the base of any deep learning beginner. This project helps us understand how neural networks work, the layers inside it, what happens if we increase the number of hidden layers or change the activation function to help us classify the hand written digit in an accurate way. This project will help you classify handwritten digits with good accuracy using Artificial neural network (ANN).

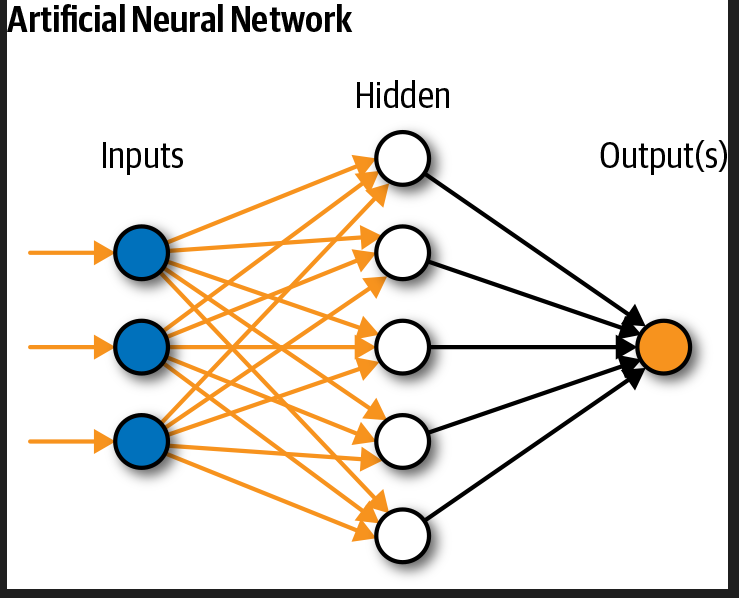
### Dataset Description:

We will be using the standard MNIST Handwritten Digit Classification Dataset which is a collection of 60,000 images of size 28x28 of handwritten digits from 0 to 9. Sample images from the dataset



**Artificial Neural Network**

ANN is a neural network which tries to perform tasks like a human does, think like a human brain. Just like a human brain understands things after learning by watching things or by experience, ANN does the same as well. It learns with experience of going through the dataset multiple times and understands the relations, hidden features and parameters. ANN is helpful in doing regression, classification tasks and performs extremely well on huge datasets achieving high accuracy.

****

**Input Layer:**

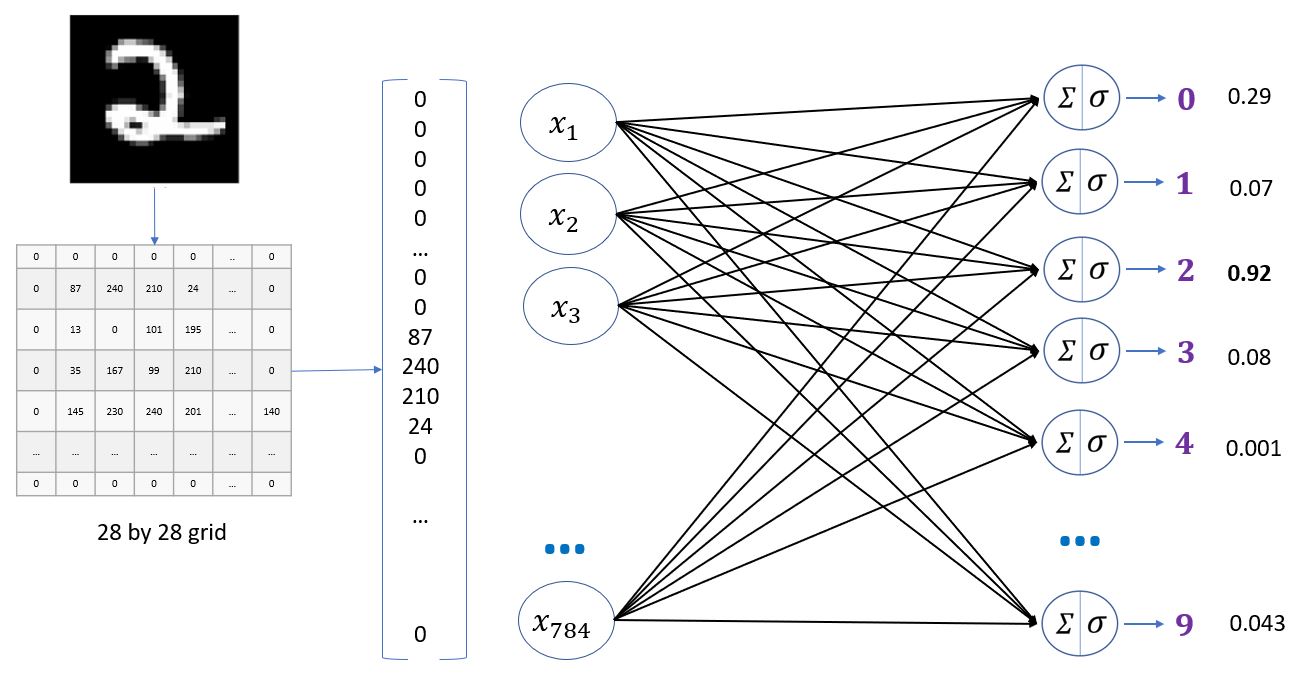
Whatever input you pass for the model to learn goes through this layer of neural network for performing calculations.

**Hidden Layer:**

The layer as the name suggests hidden because when we see the real time application we only focus on the input and output, we do not focus on how things happen. Hidden layer performs calculations, does processing, understands the hidden features and updates weights to get the best possible accuracy.

**Output Layer:**

**The input passes through hidden layer where processing happens and output is returned.**

****

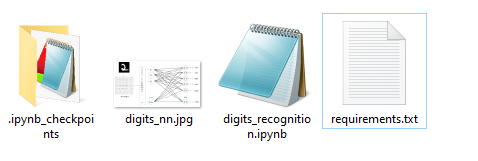
.

**How to Execute?**

So, before execution we have some pre-requisites that we need to download or install i.e., anaconda environment, python and a code editor. **Anaconda**: Anaconda is like a package of libraries and offers a great deal of information which allows a data engineer to create multiple environments and install required libraries easy and neat.

To check on how to install anaconda environment, set up jupyter notebook refer to this article. You can skip the article if you have knowledge of installing anaconda, setting up environment and installing requirements.txt

1. Install necessary libraries from requirements.txt file provided.



1. Go to the directory where your requirement.txt file is present.

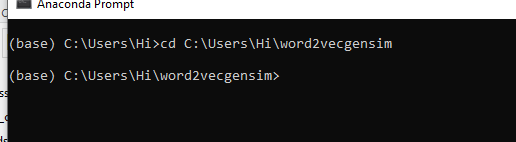
cd <<directory of your file>>. E.g, If my file is in d drive, then

1. cd d:

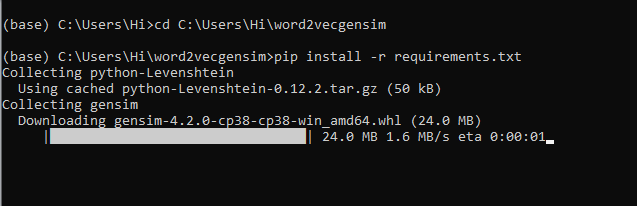
2. cd d:\License-Plate-Recognition-main  **#CHANGE PATH AS PER YOUR PROJECT, THIS IS JUST AN EXAMPLE**

If your project is in c drive, you can ignore step 1 and go with step 2.

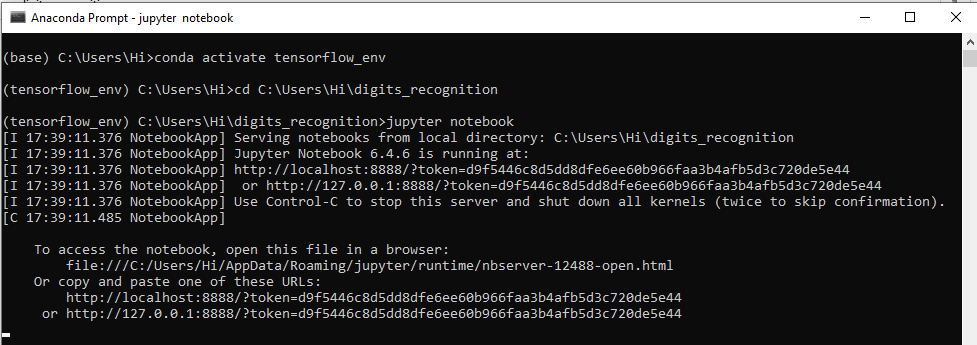
Eg. cd C:\Users\Hi\License-Plate-Recognition-main **#CHANGE PATH AS PER YOUR PROJECT, THIS IS JUST AN EXAMPLE**



1. Run command **pip install -r requirements.txt** or **conda install requirements.txt** (Requirements.txt is a text file consisting of all the necessary libraries required for executing this python file. If it gives any error while installing libraries, you might need to install them individually.)



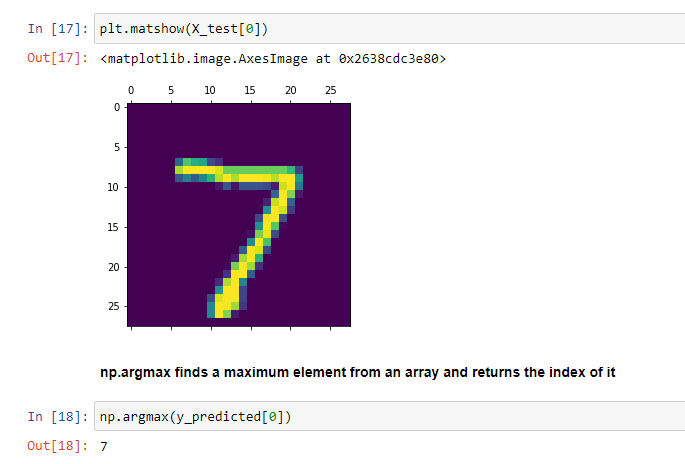
All the necessary files will get downloaded. To run the code, open anaconda prompt. Go to virtual environment if created or operate from the base itself and start jupyter notebook, open folder where your code is present.

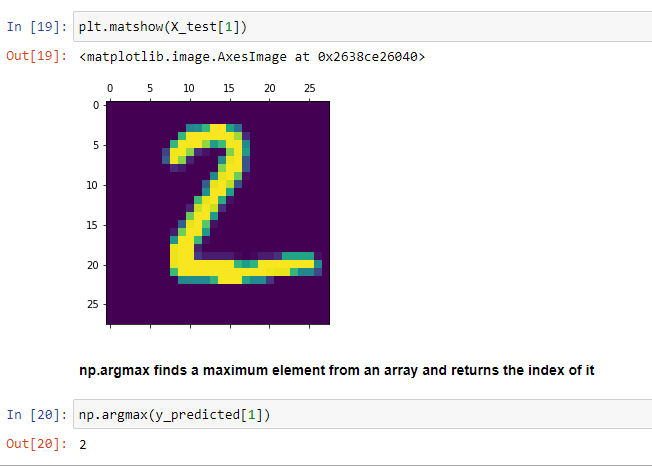
****

****

Open “digits\_recognition.ipynb” to get the results.

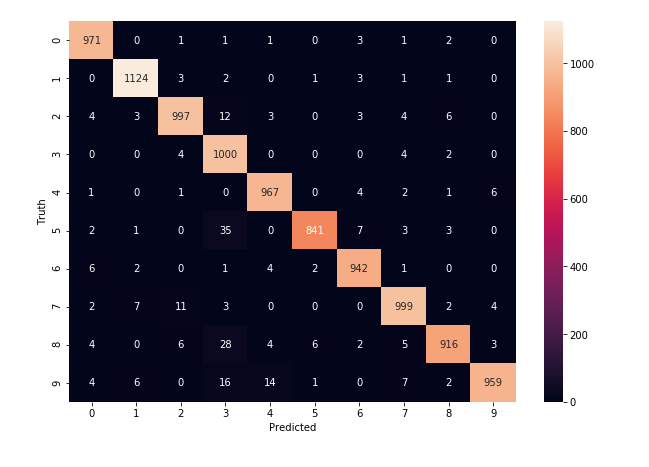
**Results:**

****

****

Results show that the model is performing well.

**Confusion Matrix for 10 classes:**



**Issues faced/ Points to note:**

1. Ensure you have all libraries installed.
2. Give correct paths wherever necessary.
3. Make sure you have the appropriate versions of tensorflow and keras.