**Introduction to Neural Network**

Bibek Lamsal

CSE student from BVC Engineering College, Amalapuram, Andhra Pradesh

**Abstract:** This paper proposes the introduction to the Neural network and its Applications.Neural networks and deep learning currently provide the best solutions to many problems in image classification and recognition, speech recognition, and natural language processing with real time learning from examples and less computation cost compare to other machine learning algorithms.

**Keywords:**

Dendrites, cell body, axon, terminal, inputs, Hidden Layer, Output, Weight, Accuracy,

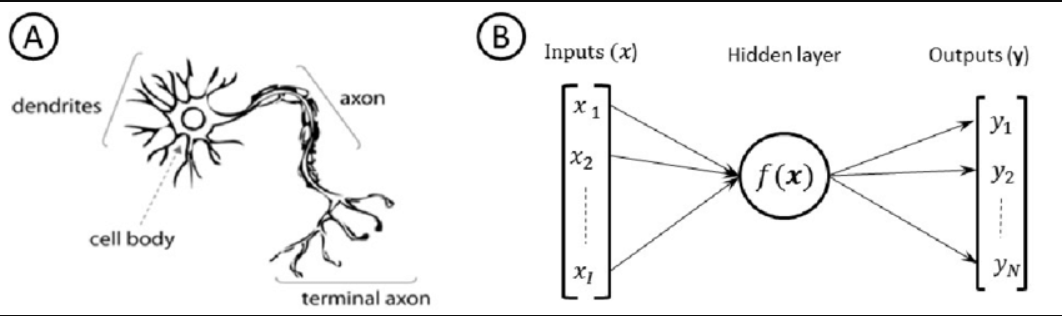
Backward Propagation, Forward Propagation, Cost function, Gradient Descent, CNN, ANN.

.

**1. Introduction:**

Neural networks, a beautiful biologically-inspired programming paradigm which enables a computer to learn from observational data. Deep learning is a powerful set of techniques for learning in neural networks. These types of algorithm learn from the examples and experience like human being learning process.

A) Neuron in the Brain B) Neuron in the Algorithm



Let’s take figure, A,

This is a neuron present in human brain where information can be stored, processed and gives output as a input to the another neuron. This stored information can be in Neuron. In each step connected neurons passes the information form one to another. Where the following things can be taken place.

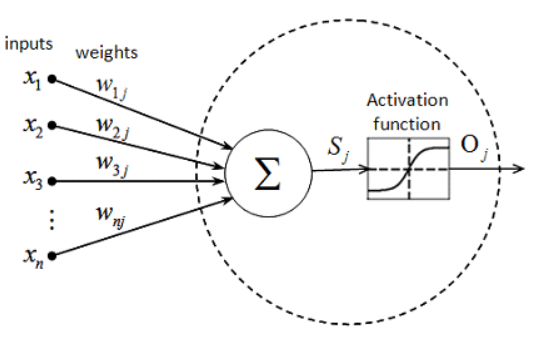
Dendrites: These are the number of input wires connected to the cells body.

Cell Body: These are the cell body where the process can be taken place.

Axon: These are the output wires connected to the next neuron.

Terminal: It is the last node which cannot be further divided.

Let’s take figure B,



Neural networks consist of neurons, connections between these neurons called **weights** and some biases connected to each neuron. We distinguish between input, hidden and output layers, where we hope each layer helps us towards solving our problem.

An important shortcoming of a perceptron is that a small change in the input values can cause a large change the output because each node (or neuron) only has two possible states: 0 or 1. A better solution would be to output a continuum of values, say any number between 0 and 1.

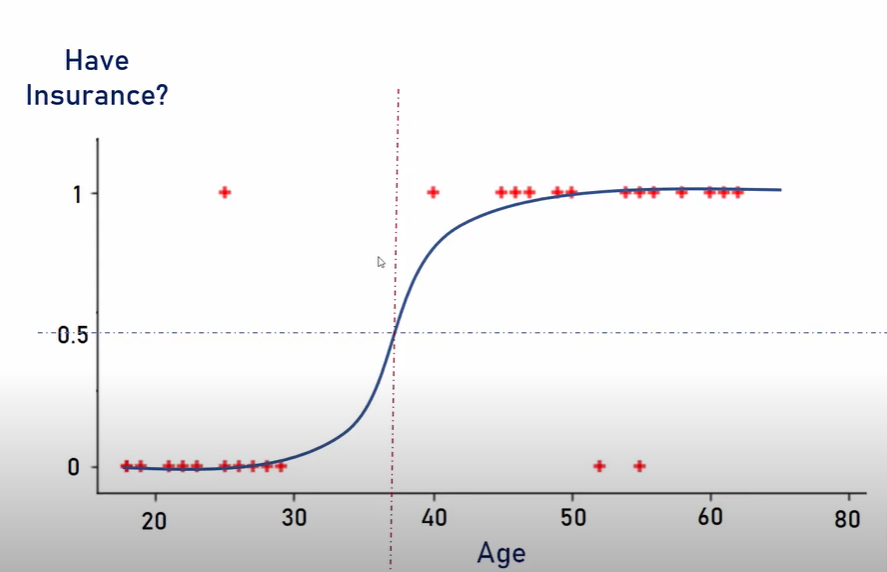
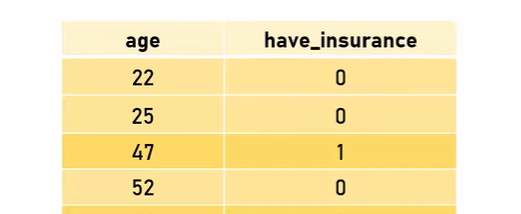
**Sigmoid Neuron:**

Sigmoid Neuron is one of the examples of neuron which gives the result in between 0 to 1.

As one option, we could simply have the neuron emit the value:

σ (w . x + b) = 1/ (1 + e−(w . x+b))

For a particularly positive or negative value of x · w + b, the result will be nearly the same as with the perceptron (i.e., near 0 or 1). For values close to the boundary of the separating hyperplane, values near 0.5 will be emitted. Let’s take example of life insurance. Where based on age, people may or not interest to do their life insurance. By using the sigmoid function, result is in the right side of the figure. This is one simple neuron whose value range is 0 to 1.



**Activation functions:**

In the sigmoid neuron example, the choice of what function to use to go from w . x + b to an output is called the activation function. Using a logistic, or sigmoid, activation function has some benefits in being able to easily take derivatives and the interpret them using logistic regression.

We now have an architecture that describes a neural network, but how do we learn the weights and bias terms in the model given a set of training data?

**Cost function:**

Cost function the primary set-up for learning neural networks is to define a cost function (also known as a loss function) that measures how well the network predicts outputs on the test set. The goal is to then find a set of weights and biases that minimizes the cost.

One example of a cost function is just squared error loss:

C(w, b) = 1/2n ∑i (yact\_i – y\_pred(xi))2

Cost function will help to evaluate the model is best or not by calculating the cost. Cost is low compare to other model means the model has high accuracy and cost is high means

The model has less accuracy.

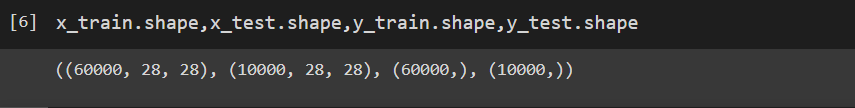
**2. Classification of digits using the Neural Networks :**

As an example of using a CNN on a real problem, we’re going to identify some handwritten numbers using the MNIST data set.

A CNN uses filters on the raw pixel of an image to learn details pattern compare to global pattern with a traditional neural net. To construct a CNN, you need to define:

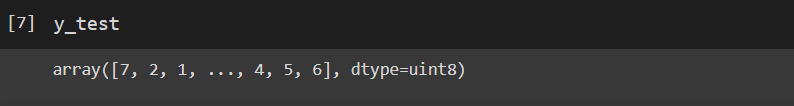
1. A convolutional layer: Apply n number of filters to the feature map. After the convolution, you need to use a Relu activation function to add non-linearity to the network.
2. Pooling layer: The next step after the convolution is to down sample the feature max. The purpose is to reduce the dimensionality of the feature map to prevent overfitting and improve the computation speed. Max pooling is the conventional technique, which divides the feature maps into subregions (usually with a 2x2 size) and keeps only the maximum values.
3. Fully connected layers: All neurons from the previous layers are connected to the next layers. The CNN will classify the label according to the features from the convolutional layers and reduced with the pooling layer.

This dataset has 60,000 rows and 784 columns for training dataset and 10,000 rows and 784 columns for the testing dataset. This dataset contains the pixel point value in each picture.



And are in the form of NumPy array.

Output also called target are in [0, 1, 2, …. , 9] range.



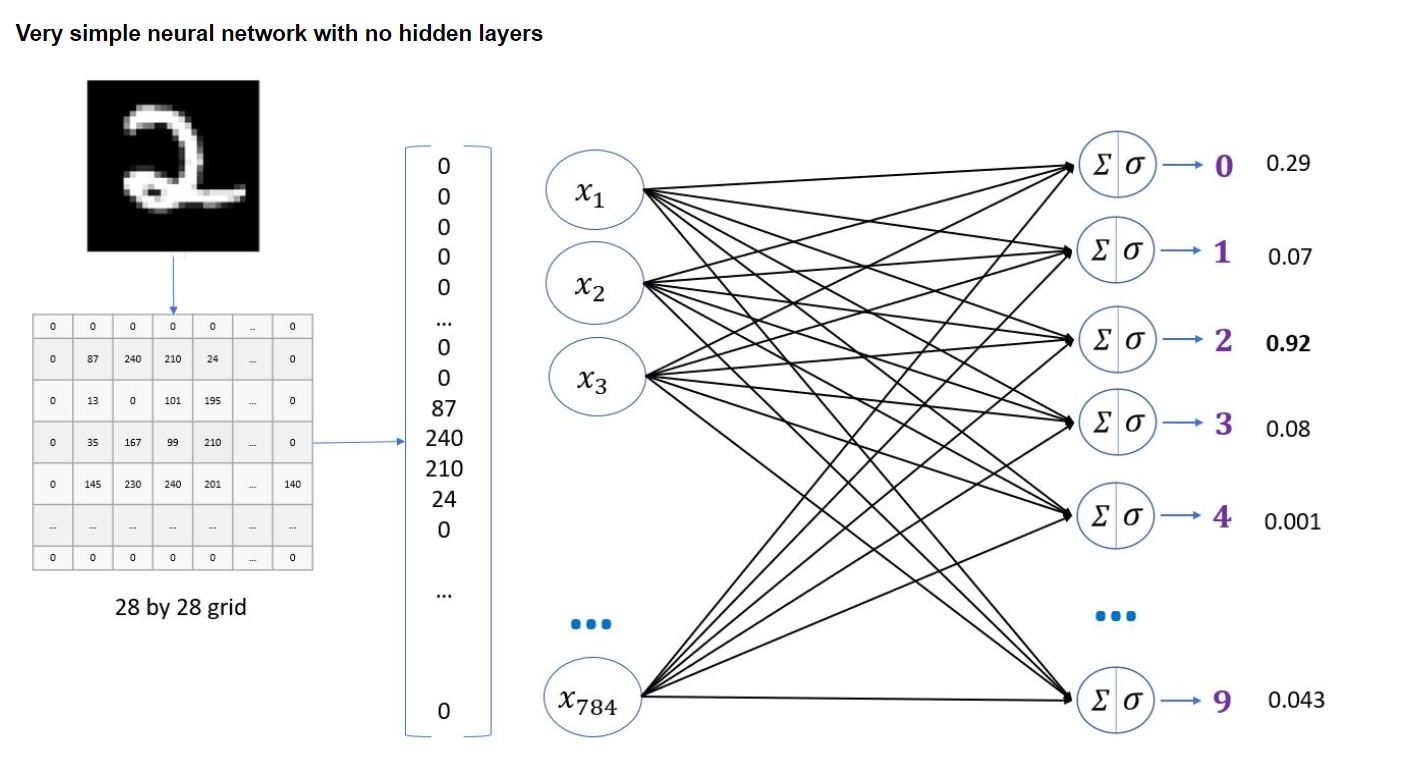
To determine which class to put a particular input into, we look at which of the output neurons have the largest output. If the output neurons have the largest value for 0 digit then we predict this is a 0 digit and vice versa. For this type of image classification Convolutional Neural Network (CNN) is best choice compare to other machine learning algorithms.

A convolutional neural network is a specific kind of neural network with multiple layers. It processes data that has a grid-like arrangement then extracts important features. One huge advantage of using CNNs is that you don't need to do a lot of pre-processing on images.

In this we take 28\*28 pixels handwritten images for training the model. Which is 784 number of total points in the images. Each of the pixel 1, 2, 3, … are taken as an input feature to the CNN.

After processing and adjusting the weights based on the hidden layer it will give the result and the model will be trained until it will match to the correct output.

Let’s the 28\*28 pixels figure and it’s associated image matrix is shown below.



**3. Proposed Approach :**

Step 1: Import the modules and dataset

Step 2: Process the data

Step 3: Create and Compile the model

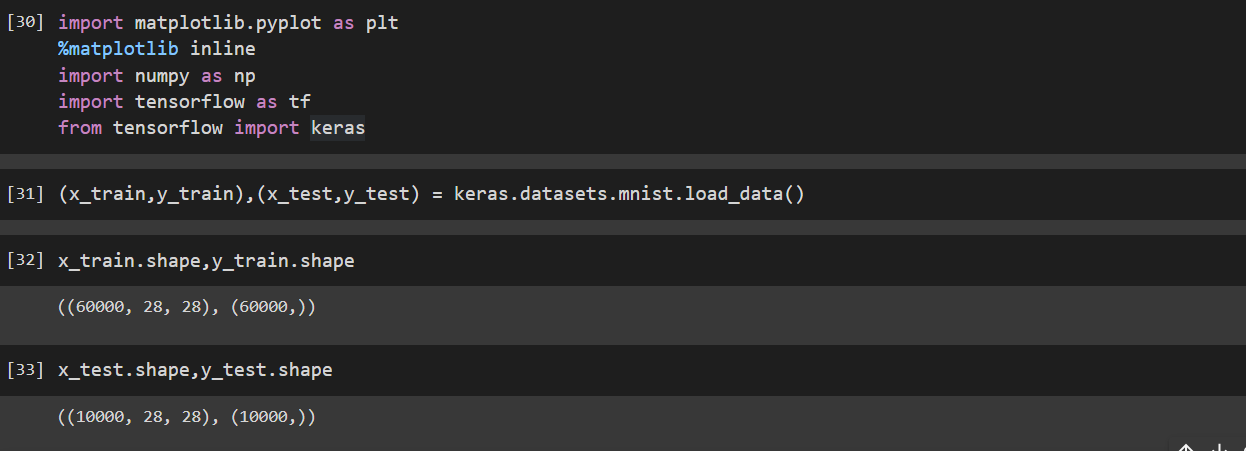
Step 4: Train the model

Step 5: Evaluate the model

Step 6: Predict the model

### Step 1: Importing Dataset

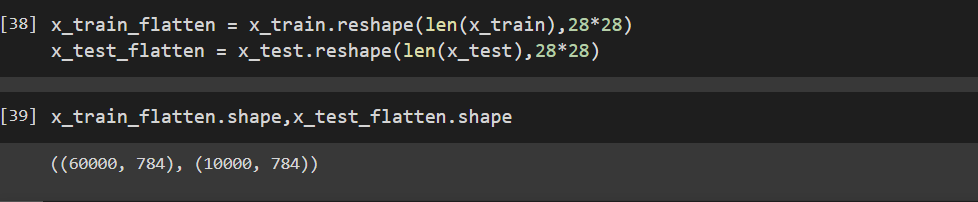
The MNIST dataset is available with scikit learn library.



Where X represents the Independent variable and Y represents the Dependent variable.

Step 2: Process dataset

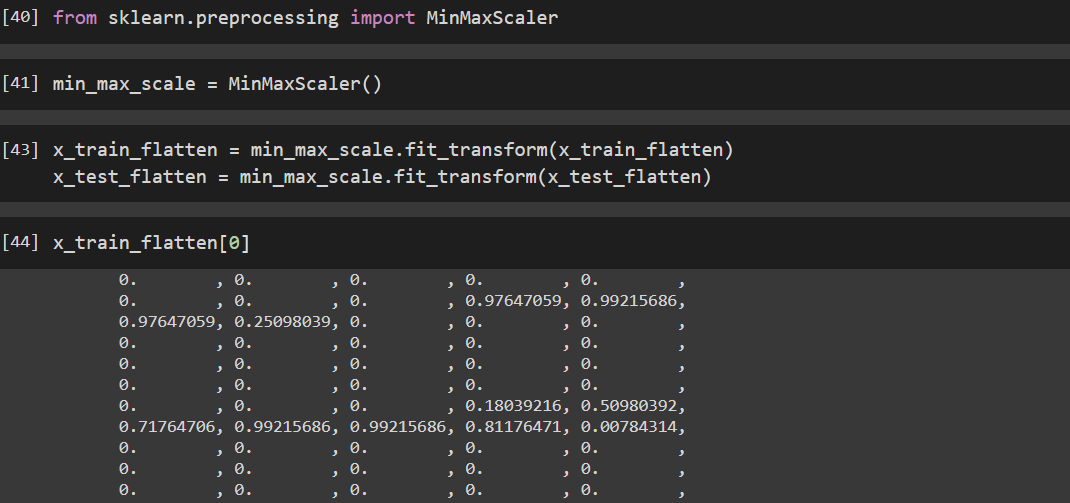
Flatten the dataset X to the 784 pixels input vector.



For getting the best result from the dataset we have to make the dataset range from 0 to 1.

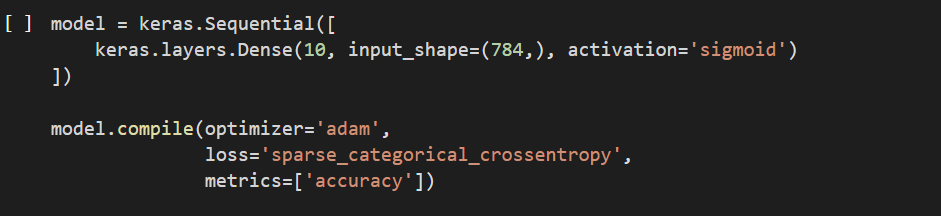
**Scale the features**

Finally, you can scale the feature with MinMaxScaler as shown in the below image classification using TensorFlow CNN example.



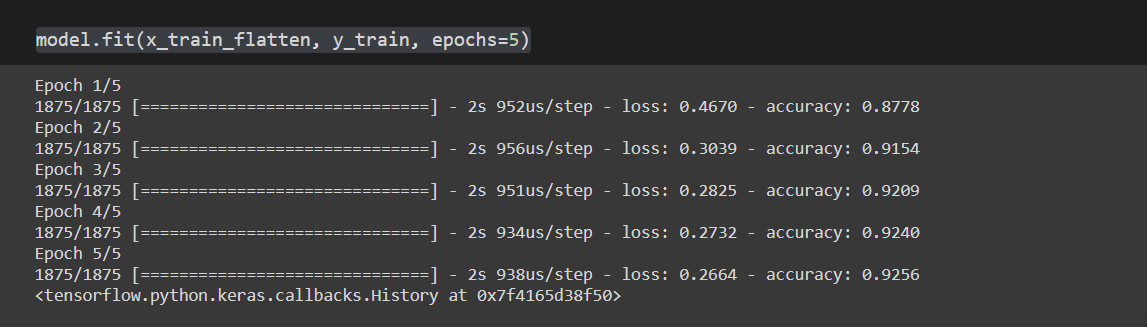
### Step 3: Create and Compile the Model

Let us create the model with input layer of 784 input elements and output layers of 10 elements.



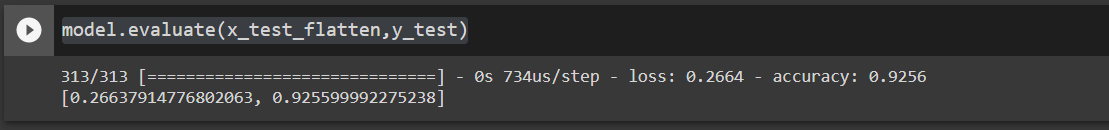
### Step 4: Train the Mode

Model can be trained by using the fit method.

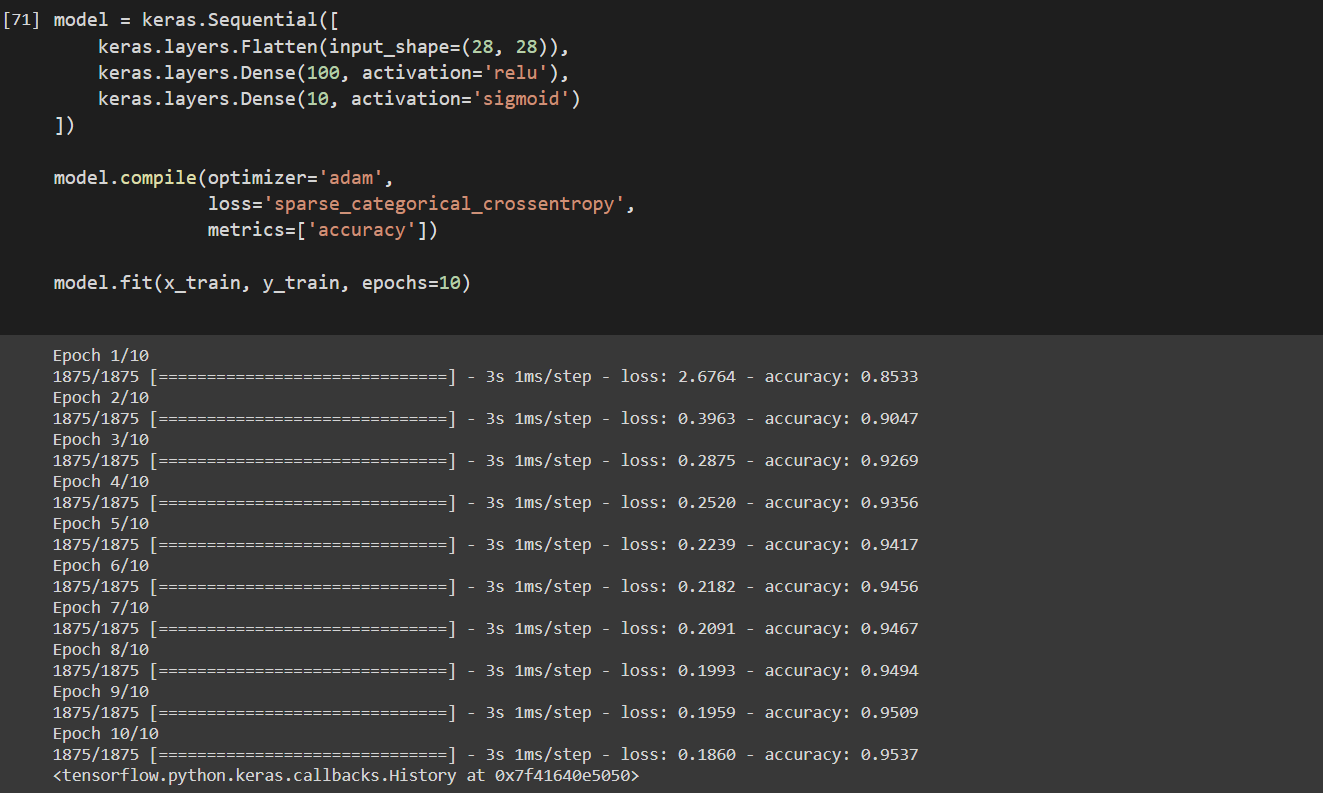


### Step 5: Evaluate the Model:

Let’s evaluate the trained model by using the test dataset.

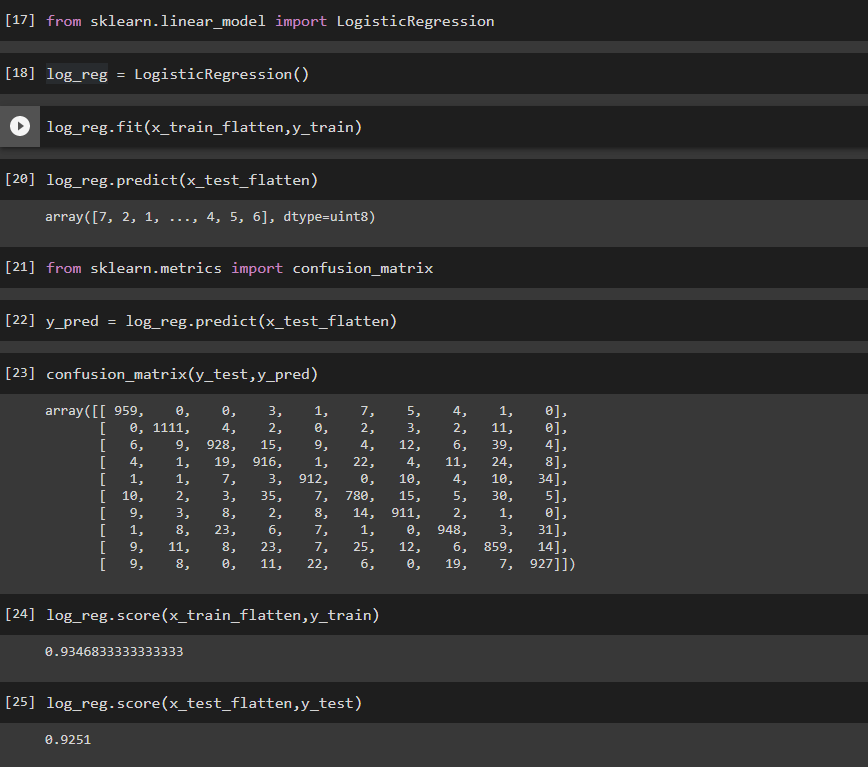
The test accuracy is 92.55%. We have created a best model to identify the handwriting digits.

### **Using Flatten layer so that we don't have to call. reshape on input dataset**



**Classification of digit using Logistic Regression**

Let’s see the result after applying the logistic regression.



1. **Summary :**

Based on the experimental analysis of logistic regression model and Conventional Neural Network (CNN) model, the better model is the CNN with accuracy of 95.37%.CNN model is still learning model where we have to change the epochs value. Neural Networks are very helpful in this type of classification problem when the input features are in higher dimension like 10’s,100’s,.. number of features are present. Where high computational cost and time is required to build a model.

The world is now moving towards fully digitalization, each and every second data is produces and stores in varying location. More data means training and testing data is also high. There is enough data to train the models using the neural network and model will be ready enough to solve the upcoming data situation not only for images others also like image recognition, speech recognition, and natural language processing with real time learning process.

**Reference:**

<http://neuralnetworksanddeeplearning.com/>

<https://www.coursera.org/learn/machine-learning>

<https://github.com/codebasics/deep-learning-keras-tf-tutorial/blob/master/1_digits_recognition/digits_recognition_neural_network.ipynb>