**CS6005-DEEP LEARNING**

**CNN IMAGE CLASSIFICATION PROJECT**

**IMAGE CLASSIFICATION OF CIFAR 10 DATASET USING CNN**

NAME: S.Karthickraja

ROLL NO:2018103547

DATE:20/04/2021

**PROBLEM STATEMENT:**

The [CIFAR-10 Dataset](https://www.cs.toronto.edu/~kriz/cifar.html) is an important image classification dataset. It consists of 60000 32x32 color images in 10 classes (airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks), with 6000 images per class. There are 50000 training images and 10000 test images.

The **GOALS** of this project are to:

* Implement different Convolutional Neural Networks (CNN) classifiers using Keras API.

**DATASET DESCRIPTION:**

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.  
  
The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.  
  
Here are the classes in the dataset, as well as 10 random images from each:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **airplane** |  |  |  |  |  |  |  |  |  |  |
| **automobile** |  |  |  |  |  |  |  |  |  |  |
| **bird** |  |  |  |  |  |  |  |  |  |  |
| **cat** |  |  |  |  |  |  |  |  |  |  |
| **deer** |  |  |  |  |  |  |  |  |  |  |
| **dog** |  |  |  |  |  |  |  |  |  |  |
| **frog** |  |  |  |  |  |  |  |  |  |  |
| **horse** |  |  |  |  |  |  |  |  |  |  |
| **ship** |  |  |  |  |  |  |  |  |  |  |
| **truck** |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

The classes are completely mutually exclusive. There is no overlap between automobiles and trucks. "Automobile" includes sedans, SUVs, things of that sort. "Truck" includes only big trucks. Neither includes pickup trucks.

**MODULES:**

* Loading the dataset
* Normalizing the training data
* Build simple ANN and see the accuracy
* Building CNN
* Prediction

**CNN MODEL SUMMARY:**

Filters : 32

Activation function used : relu

Activation function (o/p layer) : Softmax

Optimizer : Adam

Loss Function : Sparse categorical cross entropy

The model type that we will be using is **Sequential**. Sequential is the easiest way to build a model in Keras. It allows you to build a model layer by layer.

We use the ‘add()’ function to add layers to our model.

Our first 2 layers are Conv2D layers. These are convolution layers that will deal with our input images, which are seen as 2-dimensional matrices.

32 in the first layer and 64 in the second layer are the number of nodes in each layer. This number can be adjusted to be higher or lower, depending on the size of the dataset. In our case, 64 and 32 work well, so we will stick with this for now.

Kernel size is the size of the filter matrix for our convolution. So a kernel size of 3 means we will have a 3x3 filter matrix. Refer back to the introduction and the first image for a refresher on this.

Activation is the activation function for the layer. The activation function we will be using for our first 2 layers is the ReLU, or Rectified Linear Activation. This activation function has been proven to work well in neural networks.

Our first layer also takes in an input shape. This is the shape of each input image, 32,32,1 as seen earlier on, with the 1 signifying that the images are greyscale.

In between the Conv2D layers and the dense layer, there is a ‘Flatten’ layer. Flatten serves as a connection between the convolution and dense layers.

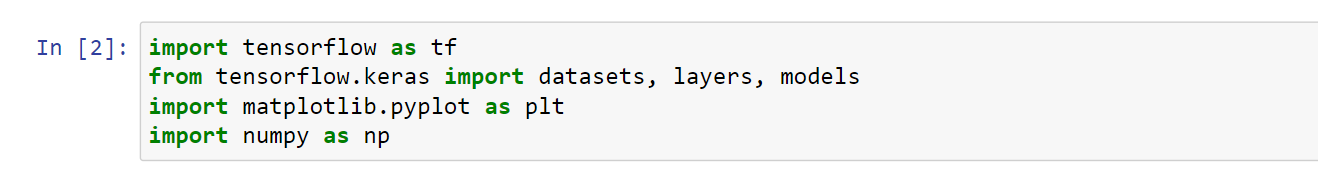
‘Dense’ is the layer type we will use in for our output layer. Dense is a standard layer type that is used in many cases for neural networks.

We will have 10 nodes in our output layer, one for each possible outcome (0–9).

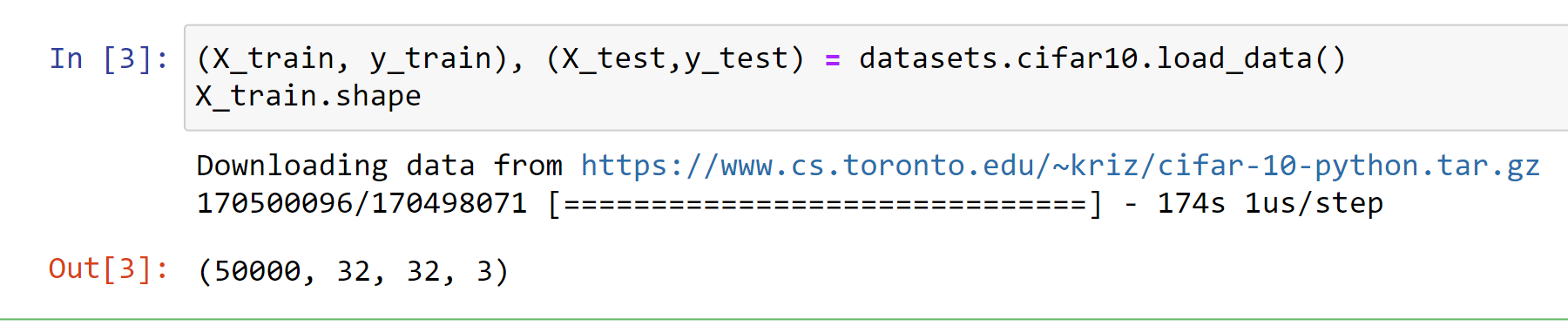
The activation is ‘softmax’. Softmax makes the output sum up to 1 so the output can be interpreted as probabilities. The model will then make its prediction based on which option has the highest probability.

**CODING SNAPSHOTS:**

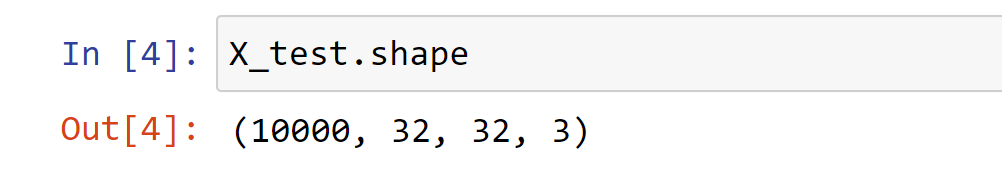
**IMPORTING LIBRARIES:**



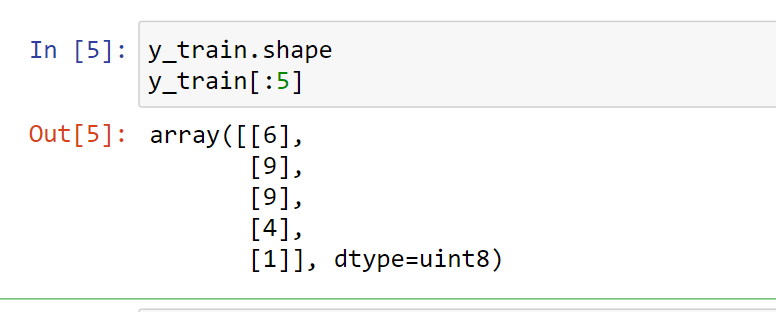
**LOADING THE DATASET:**



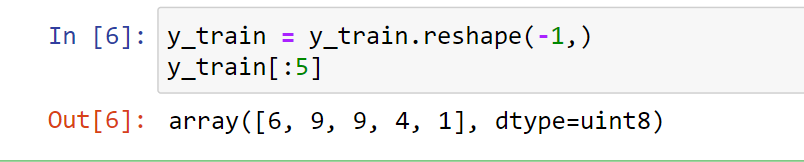
**Test dataset shape:**



**Reshaping the y\_train:**



y\_train is a 2D array, for our classification having 1D array is good enough. so we will convert this to now 1D array.



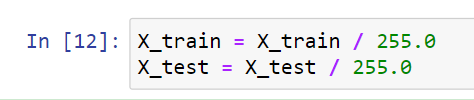


**PLOTTING IMAGES:**

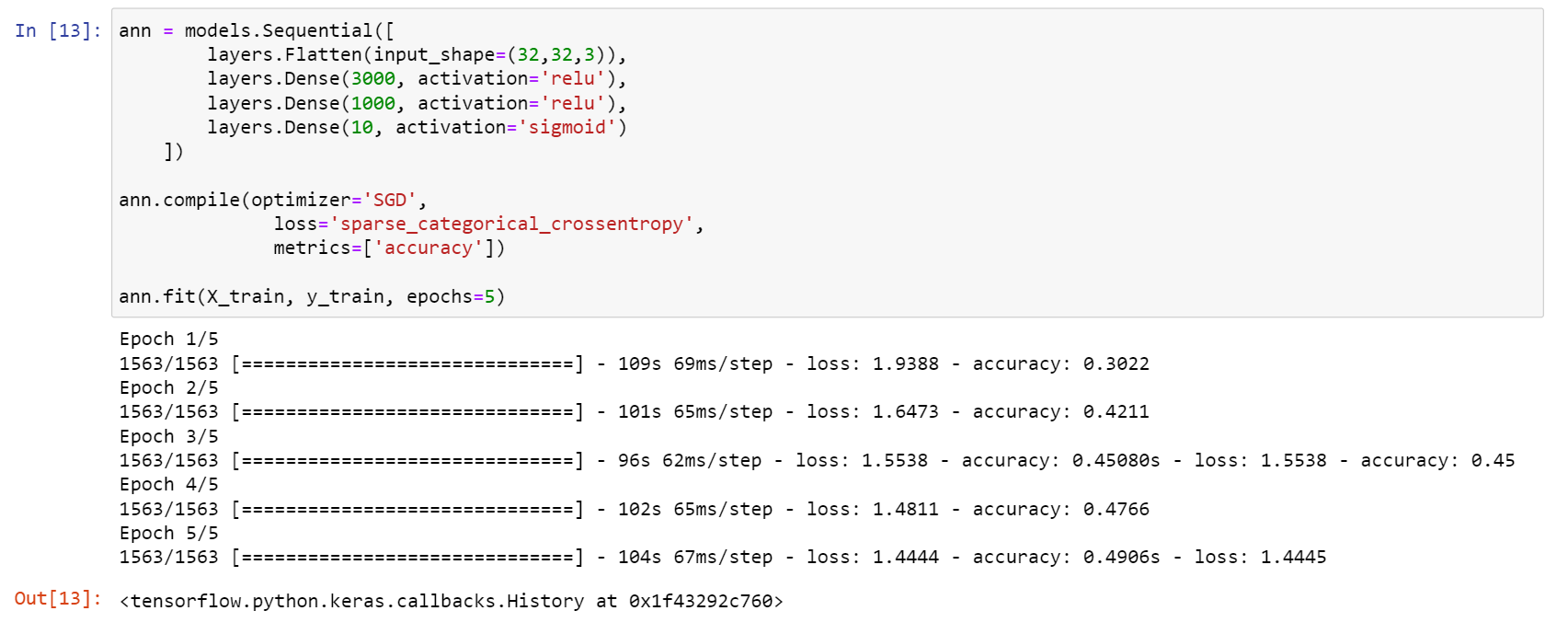


**NORMALIZING THE TRAINING DATA:**

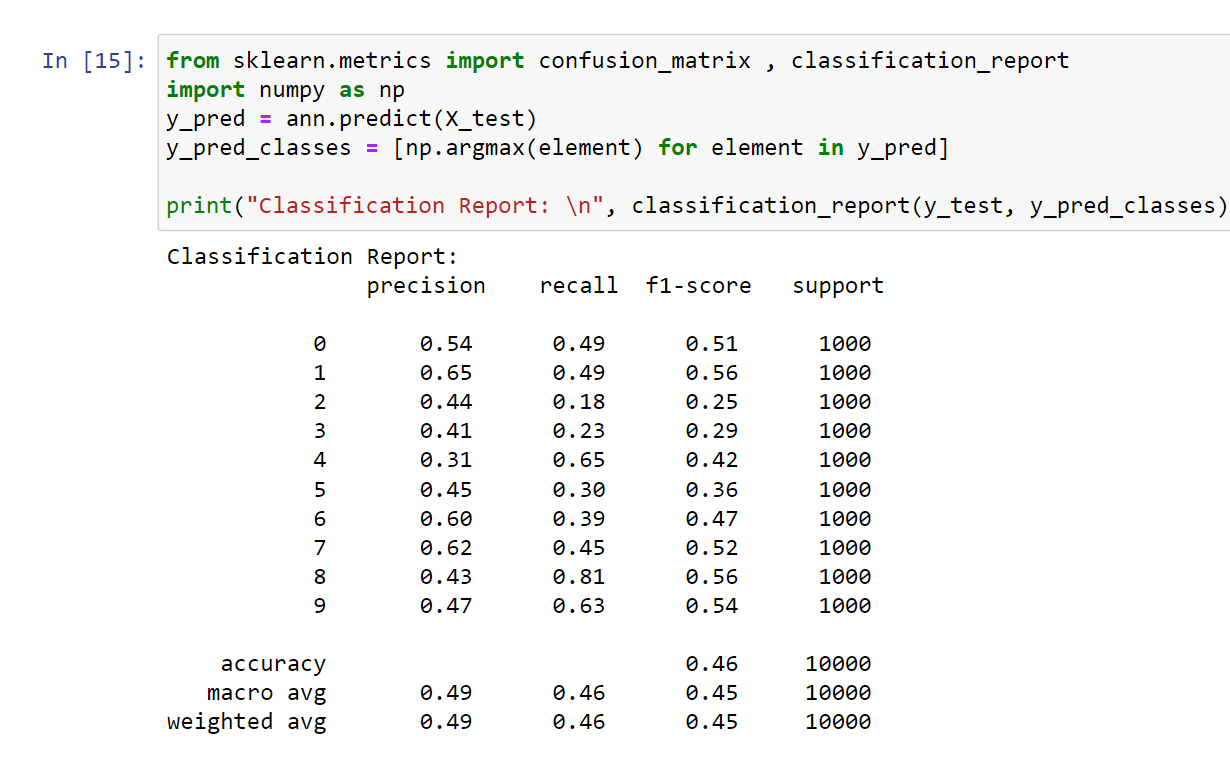
Normalize the images to a number from 0 to 1. Image has 3 channels (R,G,B) and each value in the channel can range from 0 to 255. Hence to normalize in 0-->1 range, we need to divide it by 255



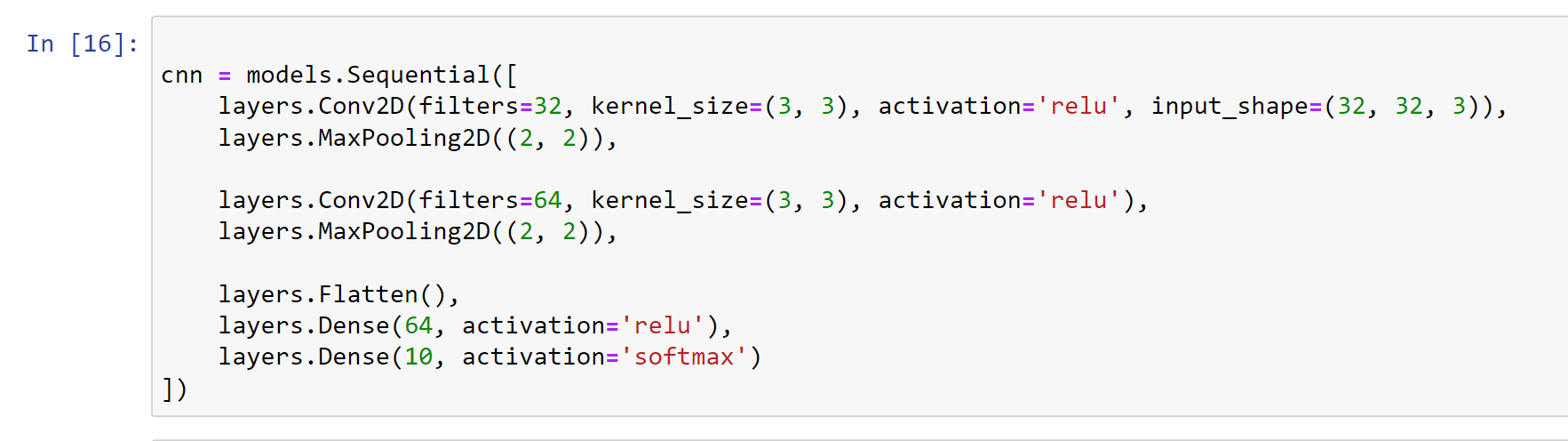
**TRAINING WITH ANN:**

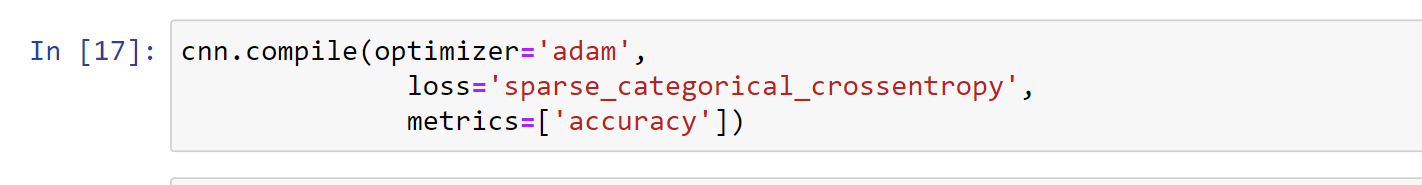


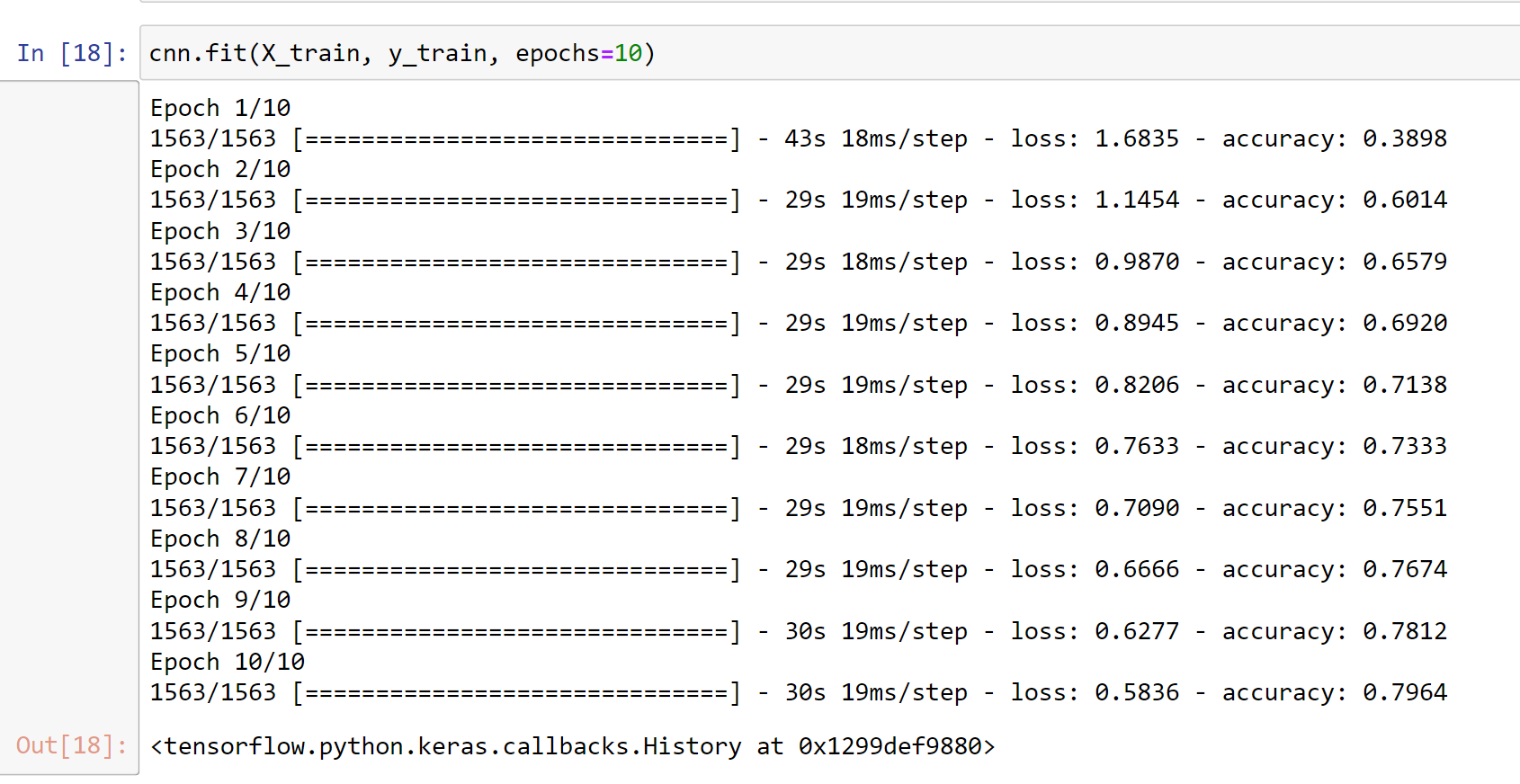
**You can see that at the end of 5 epochs, accuracy is at around 49%.Hence with ANN the accuracy is low and took long time to train.**



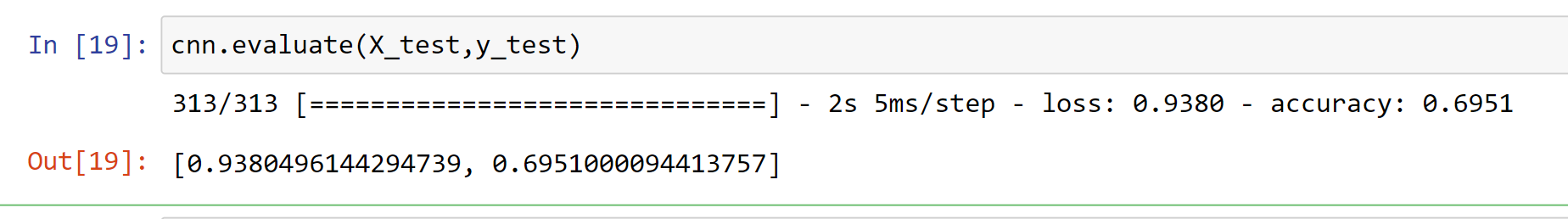
**BUILDING THE MODEL WITH CNN:**



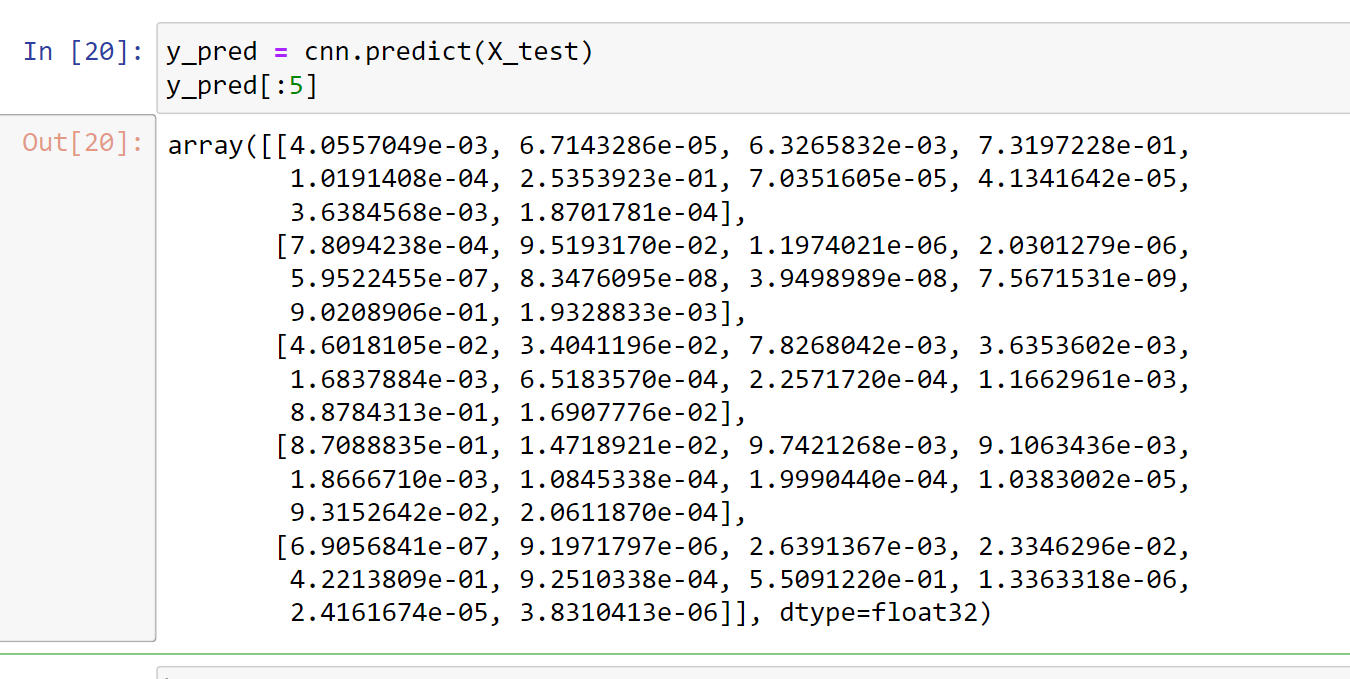


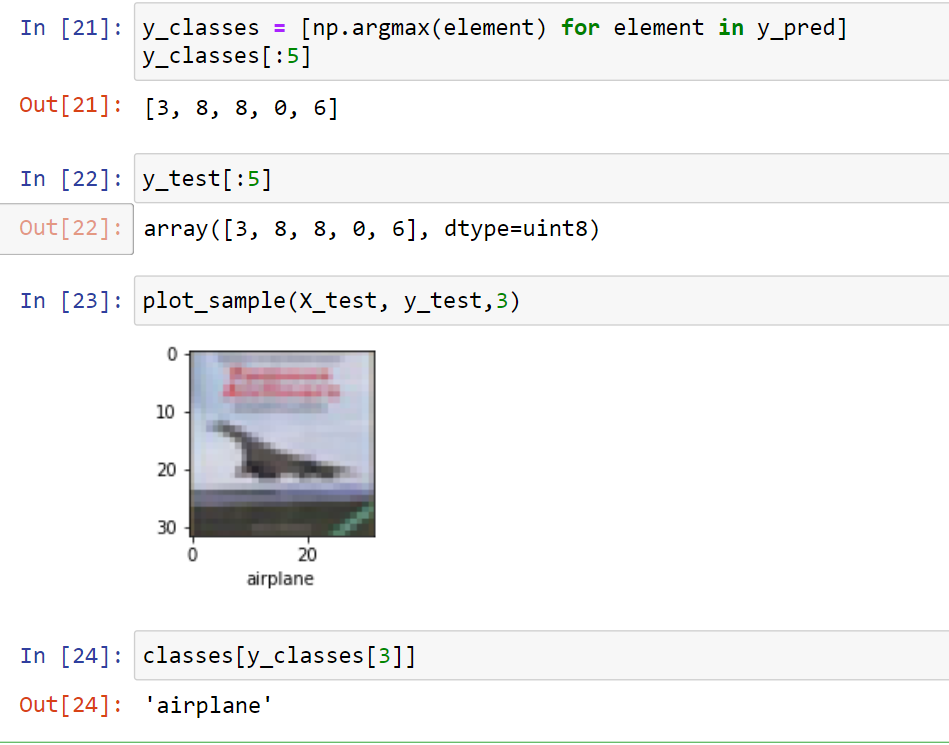


**With CNN, at the end 5 epochs, accuracy was at around 79.64% which is a significant improvement over ANN.**



**PREDICTION:**





**RESULTS:**

**Thus our CNN model successfully classified images pertaining to the corresponding classes.**

**CONCLUSION:**

**CNN's are best for image classification and gives superb accuracy. Also computation is much less compared to simple ANN as maxpooling reduces the image dimensions while still preserving the features.**

**REFERENCES:**

* <https://www.cs.toronto.edu/~kriz/cifar.html>
* <https://keras.io/api/datasets/cifar10/>
* <https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/>