Machine Learning Techniques to Classify Dogs from Cats Images

Logistic Regression :

Imported all the required libraries namely cv2 to manage images, itertools for iterations, os to parse directories/paths, pandas for data handling, sklearn for prebuilt regression and matplot for displaying images/graphs.

Next, Fetching images from train and test directory and preprocessing.

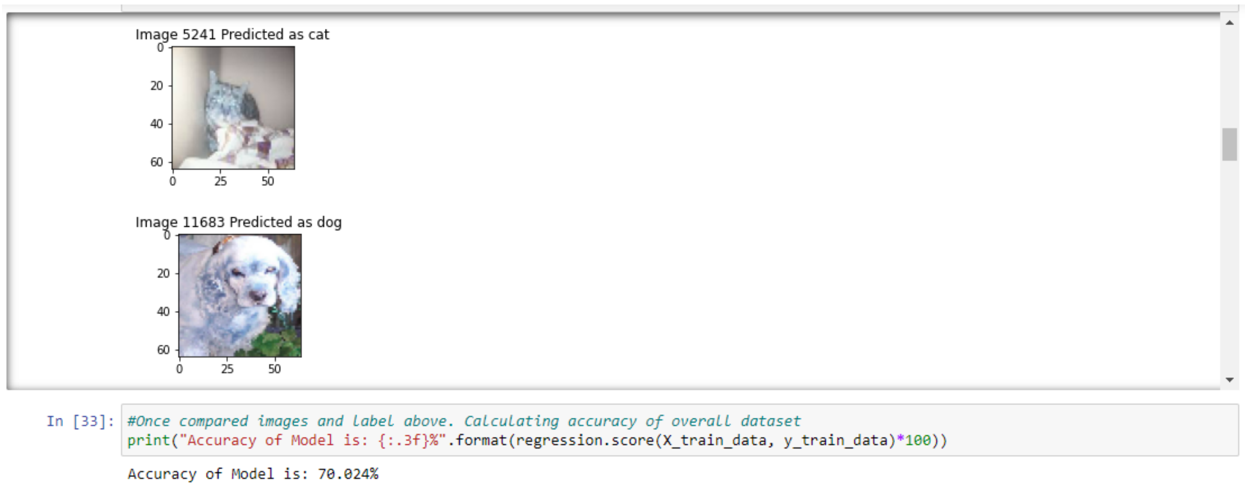
imageDataRead function to read all images in color that is why we have used 3 channels i.e R G B.

dataPreparation to generate array/matrix and bifurcating image labels as per cat(0) and dog(1)

showPredictionresult function to display the test image and its respective predicted value.

Lastly, using prebuild regression provided by sklearn to run algorithm and displaying accuracy for overall data.

Output –



```
In [33]: #Once compared images and label above. Calculating accuracy of overall dataset
         print("Accuracy of Model is: {:.3f}%".format(regression.score(X_train_data, y_train_data)*100))

         Accuracy of Model is: 70.024%
```

PyTorch :

To run Densenet model from Pytorch I have used google colab to get GPU support online.

Firstly, importing libraries and fetching data from google drive

Next transforming both train and test data as per model requirements.

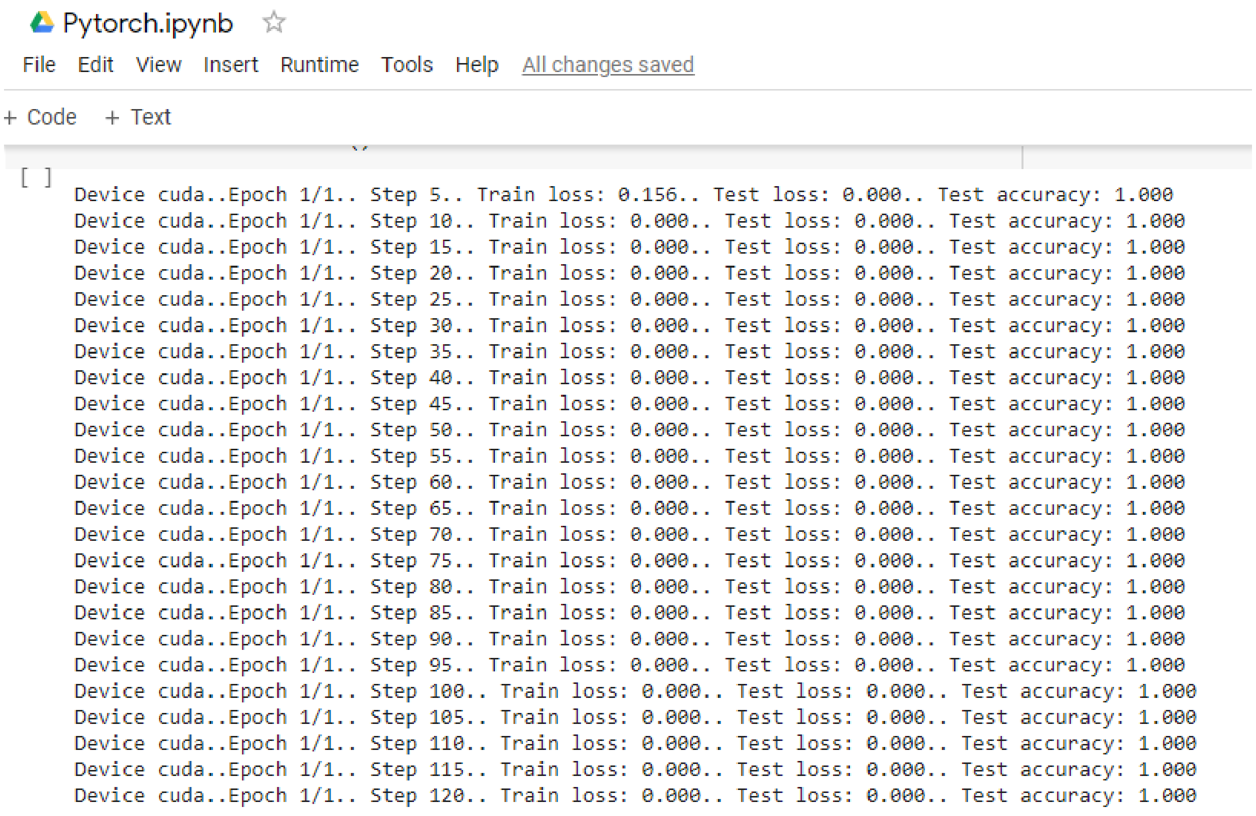
I have used densenet121 model here for higher accuracy. Had also tried resnet50.

Next, freezing parameters and handling module layers.

GPU in such models improves time complexity to much extent, Using cuda to initiate the same.

Lastly, it will run the loop forward and backward for 390 iterations as per the value of loadtrain variable and here we are printing output at every 5 steps. For each step we are calculating train loss, test loss and Accuracy of the model.

Output –



```
Device cuda..Epoch 1/1.. Step 5.. Train loss: 0.156.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 10.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 15.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 20.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 25.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 30.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 35.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 40.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 45.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 50.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 55.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 60.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 65.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 70.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 75.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 80.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 85.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 90.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 95.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 100.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 105.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 110.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 115.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
Device cuda..Epoch 1/1.. Step 120.. Train loss: 0.000.. Test loss: 0.000.. Test accuracy: 1.000
```

Conclusion :

- Accuracy Pytorch (Densenet121) – Almost 100%
- Accuracy Logistic Regression – 70%~

Though running Pytorch model takes more time to run, It provides higher accuracy when compared to Logistic regression.