**CS5590 APS - Python Programming**

**Lab Work Group**

**University of Missouri – Kansas City**

**LAB Report 1**

Andrew Knowles # 17

Shah Rukh Khan # 16

Gabriel Rhodes # 31

**Tutor:** Goudarzvand, Saria

**Deadline:** 4/30/2019

**Current Date:** 4/30/2019

1. Introduction

Laboratory problem 3 of the Course CS5590 is about Machine Learning and deep learning. The Laboratory Problem 3 consists of 6 problems, each of different nature, which cover the topics that have been taught till now in the class.

This report consists of all the codes and outputs of those codes that produce the results as desired by the instructor.

1. Objectives

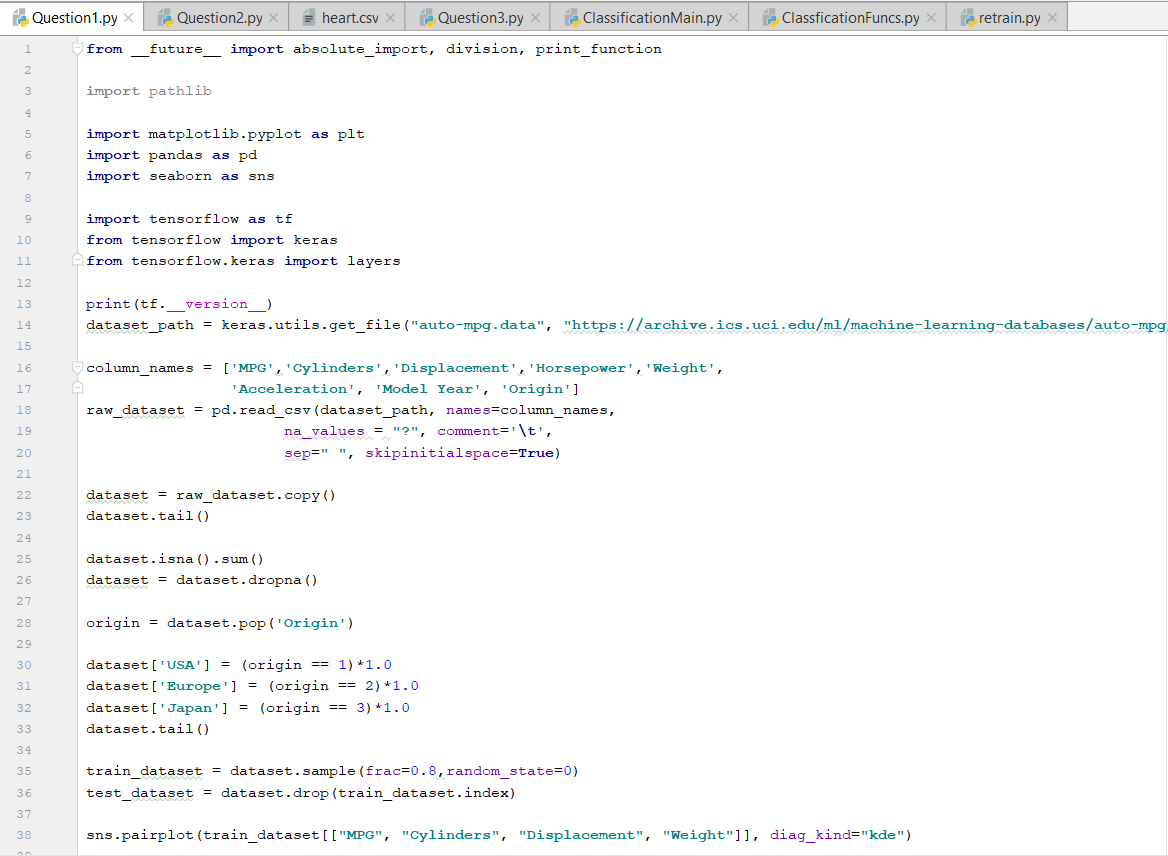
The main objective of this Laboratory 3 is to test the familiarity of the students with all that is taught till now in CS5590 class. Including but not limited to Deep learning, Machine learning and layers of Neural Network, Numpy and all of which is important towards the successful completion of the Deep Learning Tasks.

This reports objective is to Provide the code and detailed explanation of the codes written in response to each question of Lab Problem 3.

1. Approaches/Methods

The Approaches and Methods are all those that have been directed by the supervisor. She gave us the complete guidelines as to which method to follow, these were different for each question.

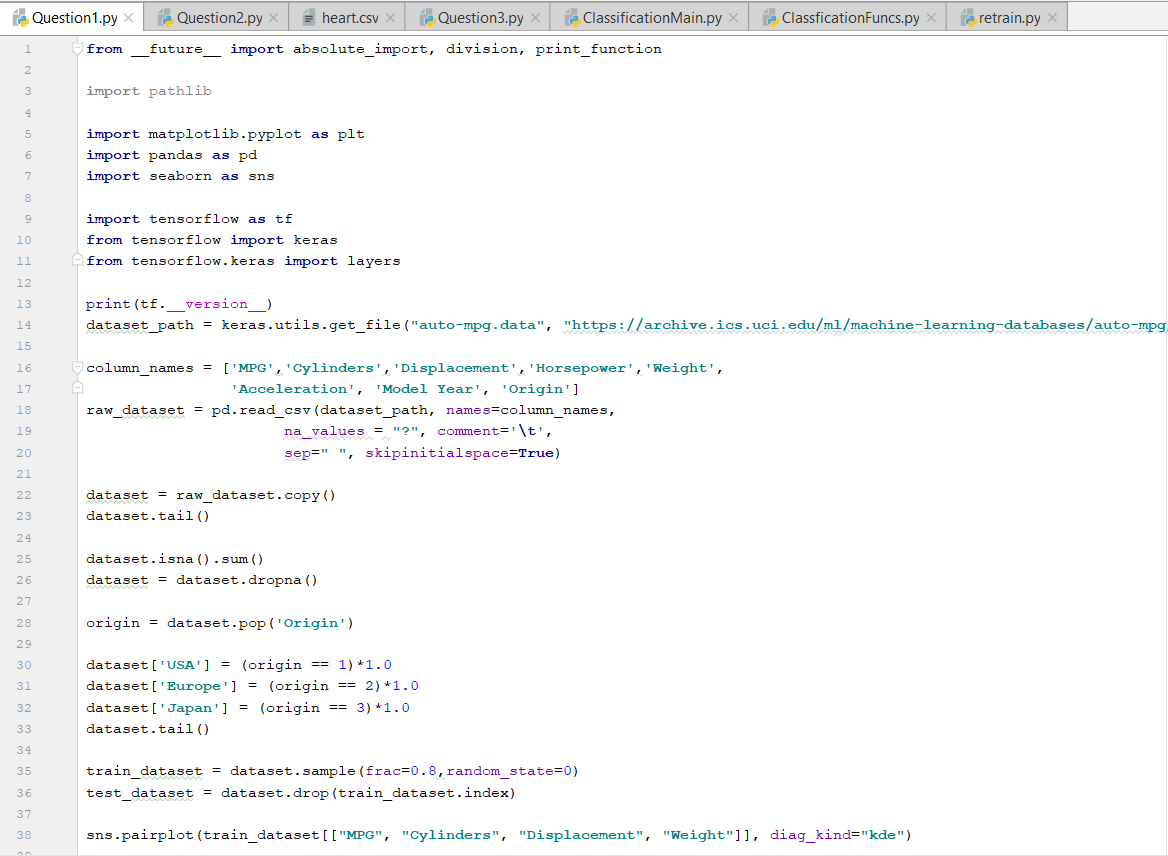
Question 1 was about Creating a Linear Regression model and predict values of a dataset that has not been used in the class. I took a Miles Per Gallon problem and computed the MPG Regression:



1. Work/Code/Answers

There are 6 Questions and all have been done in order which they were mention in the main paper. Please see below the Questions and their Complete responses.

**Question 1.**



****

****

Figure Question 1 Code

The Result of this code is Below:

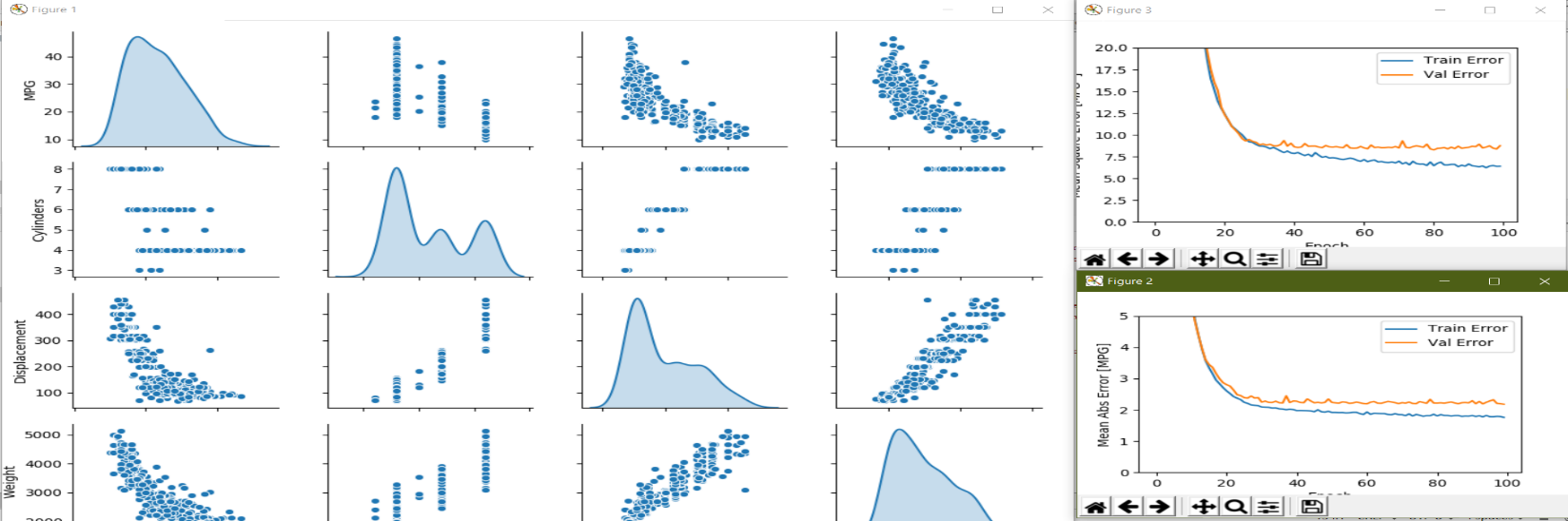
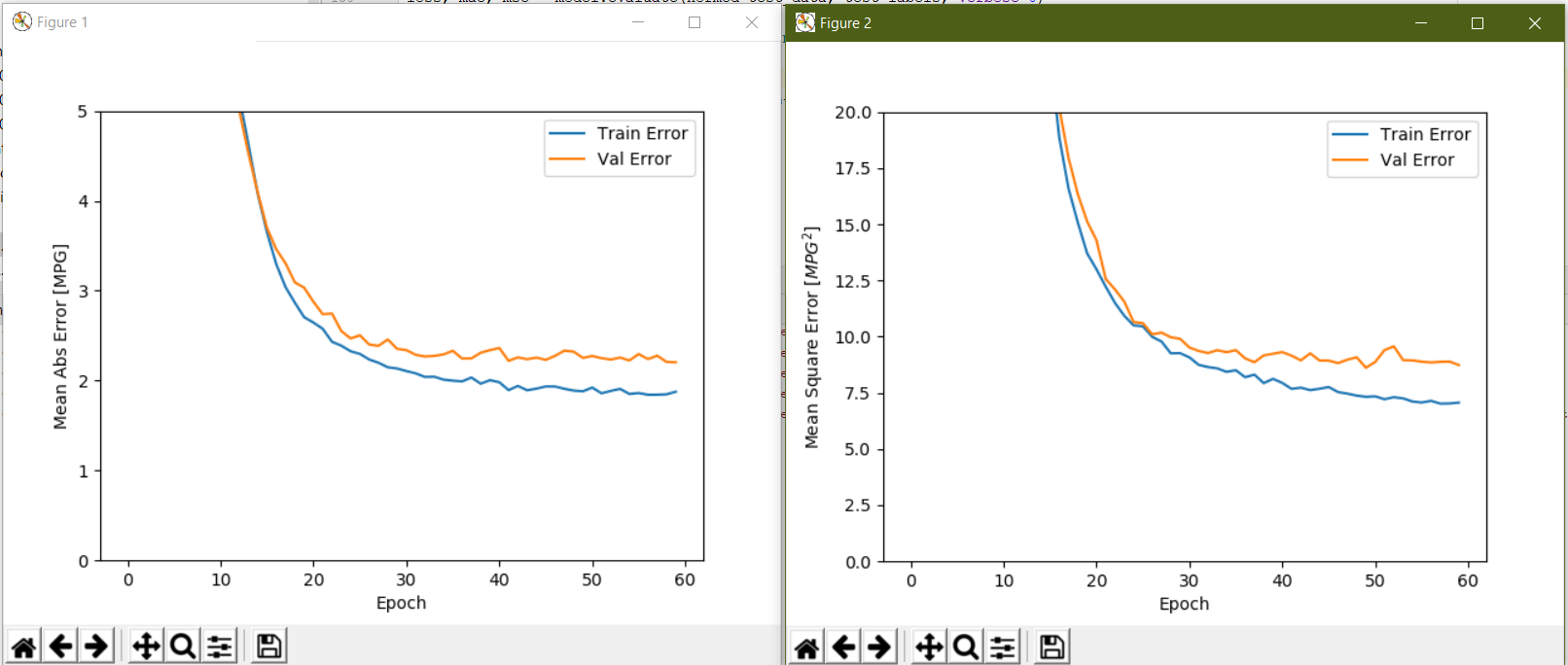


Figure Result Output of Question 1 Code



**Question 2:**

The Complete code for this Question 2 is below:

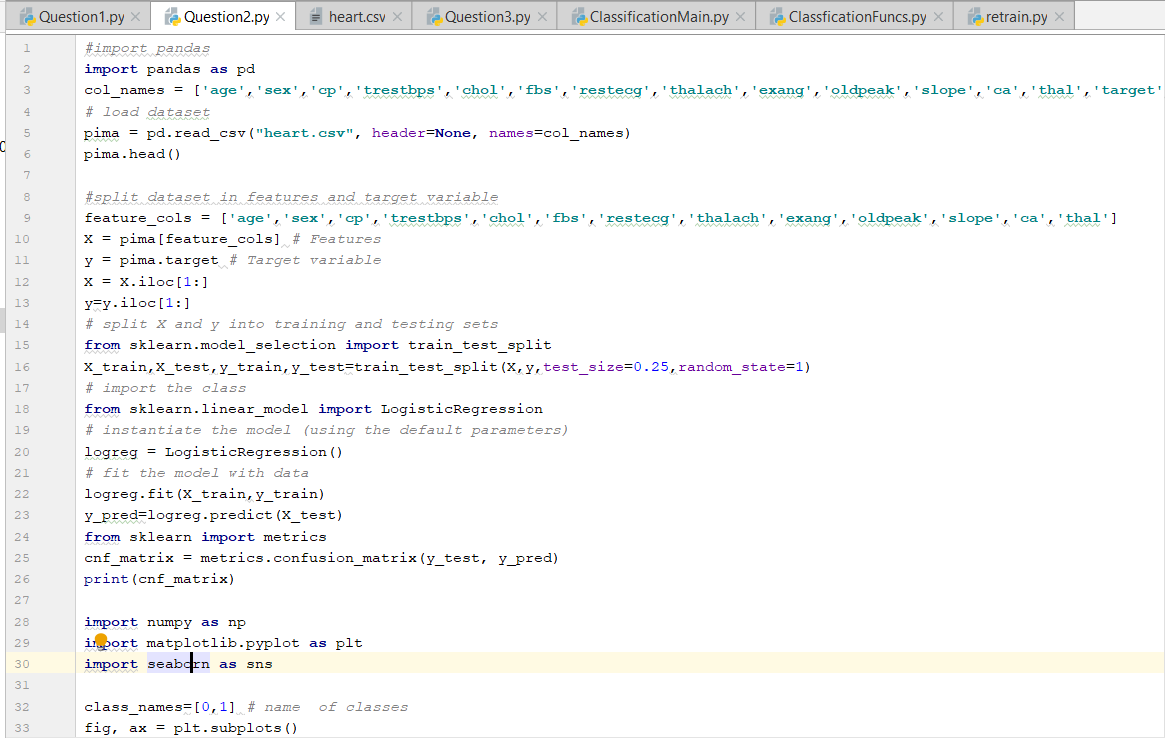


Figure Question 2 Code

The Result output for the Question 2 is below:

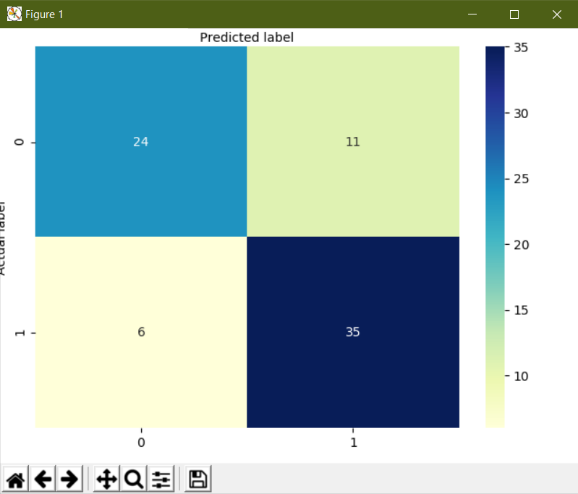
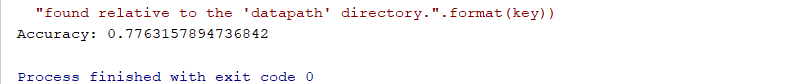


Figure Result output for Question 2 Code

****

**Question3.**

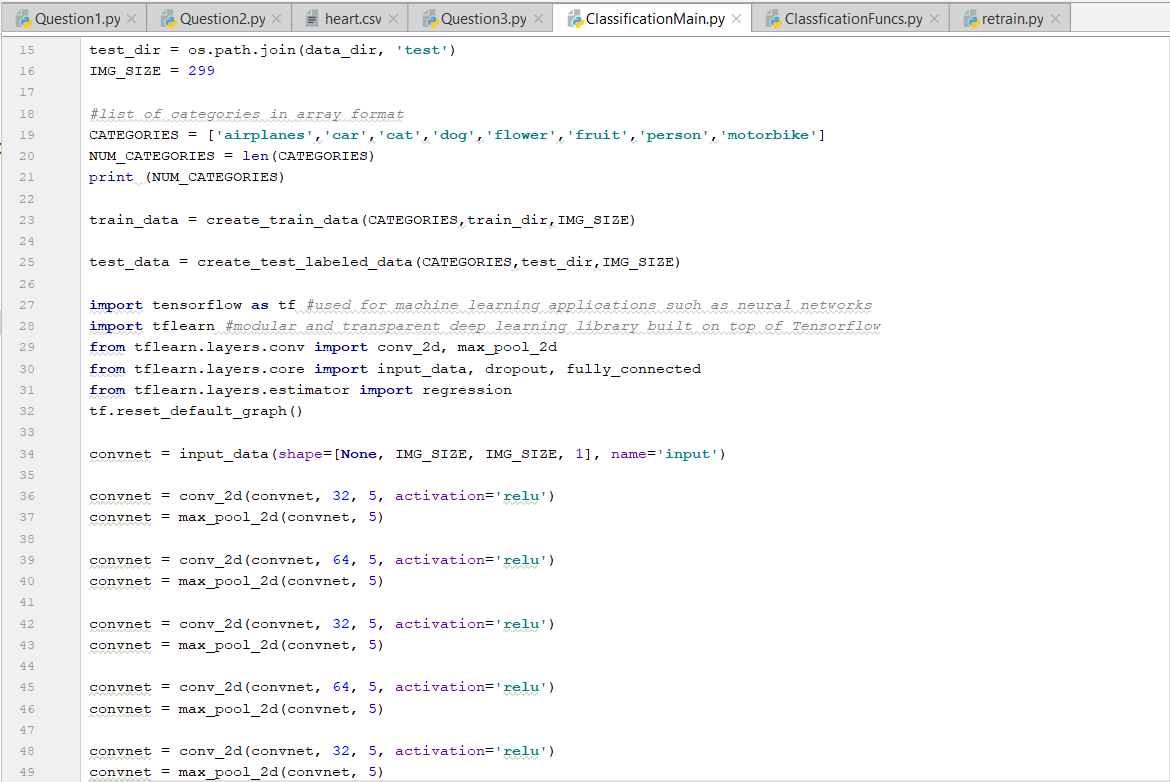
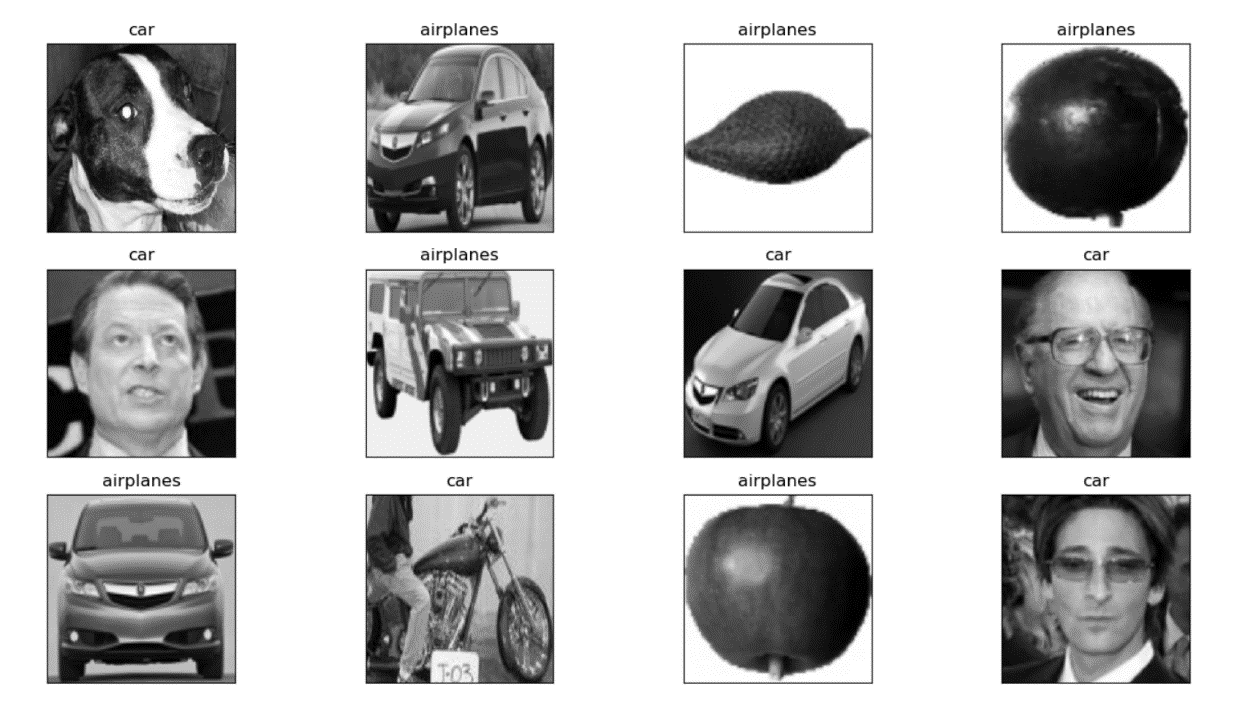


Figure Question 3 Complete Code

The Result of this Question is below:



Increasing Epochs gave better result for same model.

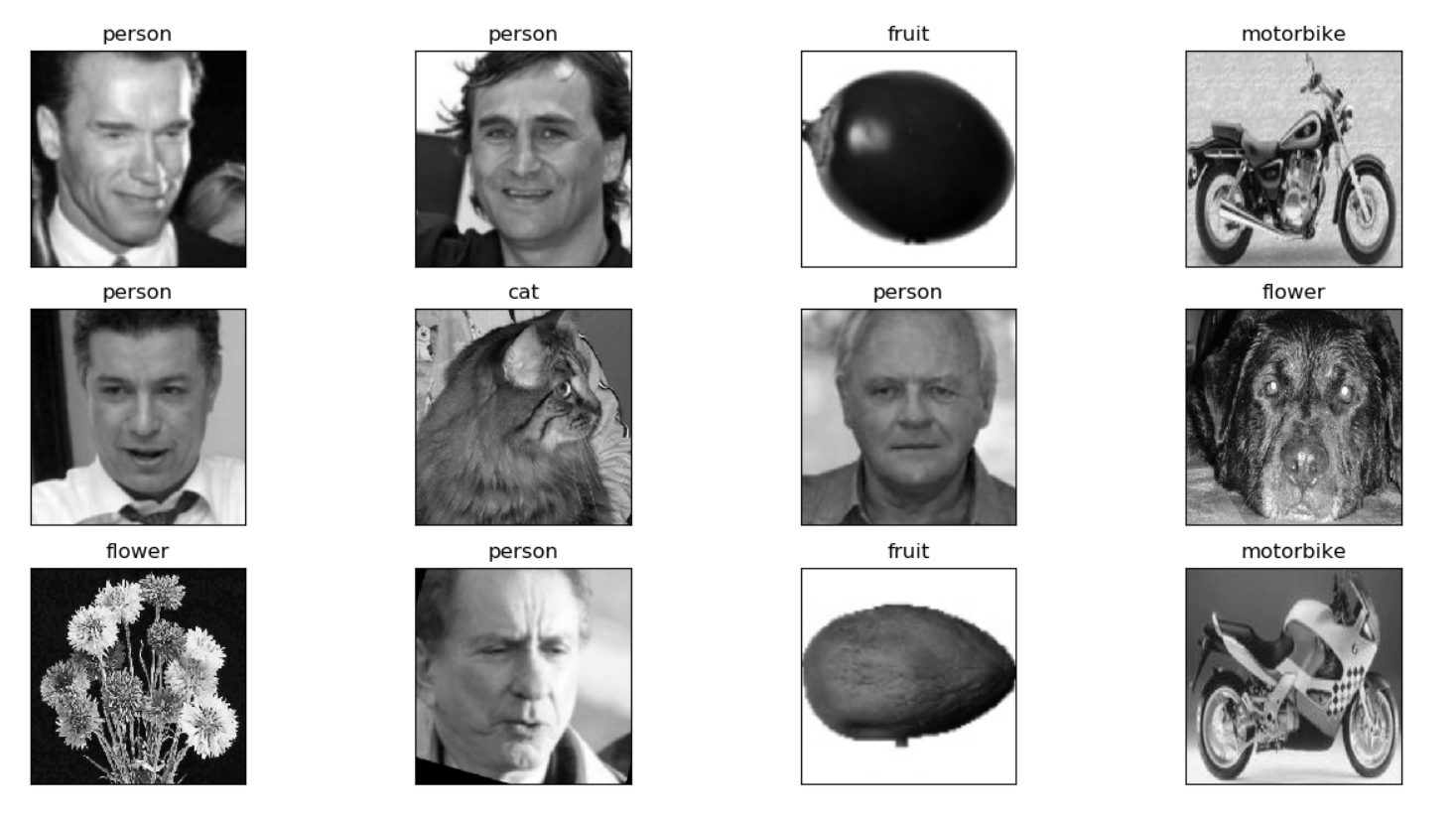


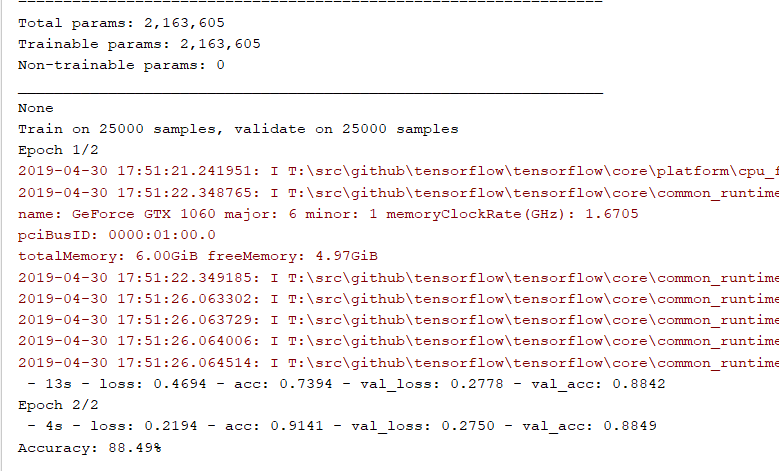
Figure Result of Question 3 Code

**Question4**

**Code:**

*#Question 4***import** numpy  
**from** keras.datasets **import** imdb  
**from** keras.models **import** Sequential  
**from** keras.layers **import** Dense  
**from** keras.layers **import** Flatten  
**from** keras.layers.convolutional **import** Conv1D  
**from** keras.layers.convolutional **import** MaxPooling1D  
**from** keras.layers.embeddings **import** Embedding  
**from** keras.preprocessing **import** sequence  
**from** keras.layers **import** LSTM  
*# fix random seed for reproducibility*seed = 7  
numpy.random.seed(seed)  
top\_words = 5000  
(X\_train, y\_train), (X\_test, y\_test) = imdb.load\_data(num\_words=top\_words)  
*# pad dataset to a maximum review length in words*max\_words = 500  
X\_train = sequence.pad\_sequences(X\_train, maxlen=max\_words)  
X\_test = sequence.pad\_sequences(X\_test, maxlen=max\_words)  
  
*# create the model*model = Sequential()  
model.add(Embedding(top\_words, 32, input\_length=max\_words))  
model.add(Conv1D(filters=32, kernel\_size=3, padding=**'same'**, activation=**'relu'**))  
model.add(MaxPooling1D(pool\_size=2))  
model.add(Flatten())  
model.add(Dense(250, activation=**'relu'**))  
model.add(Dense(1, activation=**'sigmoid'**))  
model.compile(loss=**'binary\_crossentropy'**, optimizer=**'adam'**, metrics=[**'accuracy'**])  
print(model.summary())  
  
*# create the model*model = Sequential()  
model.add(Embedding(top\_words, 32, input\_length=max\_words))  
model.add(Conv1D(filters=32, kernel\_size=3, padding=**'same'**, activation=**'relu'**))  
model.add(MaxPooling1D(pool\_size=2))  
model.add(Flatten())  
model.add(Dense(250, activation=**'relu'**))  
model.add(Dense(1, activation=**'sigmoid'**))  
model.compile(loss=**'binary\_crossentropy'**, optimizer=**'adam'**, metrics=[**'accuracy'**])  
print(model.summary())  
  
*# Fit the model*model.fit(X\_train, y\_train, validation\_data=(X\_test, y\_test), epochs=2, batch\_size=128, verbose=2)  
*# Final evaluation of the model*scores = model.evaluate(X\_test, y\_test, verbose=0)  
print(**"Accuracy: %.2f%%"** % (scores[1]\*100))

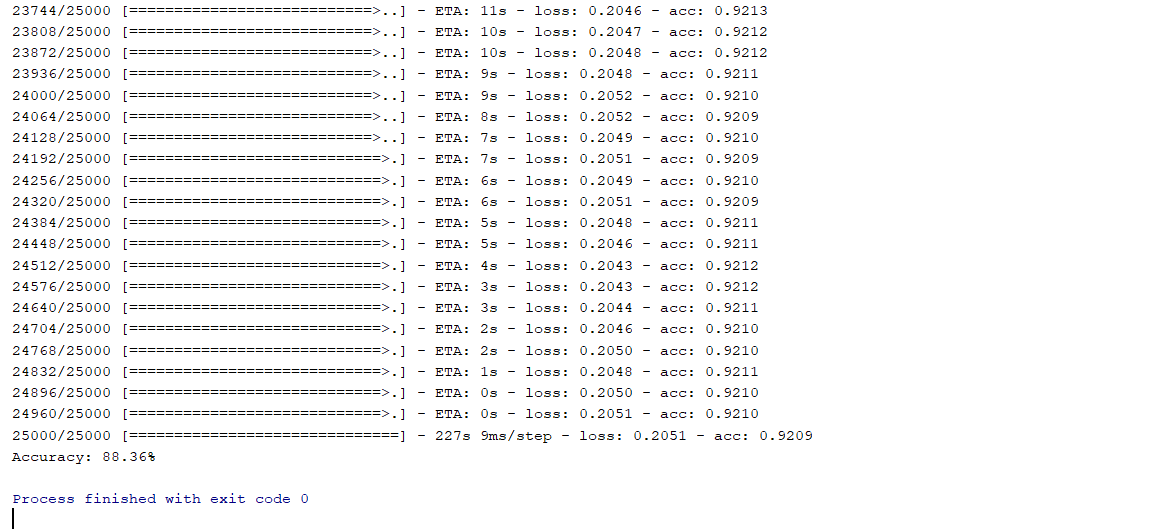
**Output**

****

**Question5**

*#Question 5  
# LSTM and CNN for sequence classification in the IMDB dataset  
# fix random seed for reproducibility*numpy.random.seed(7)  
*# load the dataset but only keep the top n words, zero the rest*top\_words = 5000  
(X\_train, y\_train), (X\_test, y\_test) = imdb.load\_data(num\_words=top\_words)  
*# truncate and pad input sequences*max\_review\_length = 500  
X\_train = sequence.pad\_sequences(X\_train, maxlen=max\_review\_length)  
X\_test = sequence.pad\_sequences(X\_test, maxlen=max\_review\_length)  
*# create the model*embedding\_vecor\_length = 32  
model = Sequential()  
model.add(Embedding(top\_words, embedding\_vecor\_length, input\_length=max\_review\_length))  
model.add(Conv1D(filters=32, kernel\_size=3, padding=**'same'**, activation=**'relu'**))  
model.add(MaxPooling1D(pool\_size=2))  
model.add(LSTM(100))  
model.add(Dense(1, activation=**'sigmoid'**))  
model.compile(loss=**'binary\_crossentropy'**, optimizer=**'adam'**, metrics=[**'accuracy'**])  
print(model.summary())  
model.fit(X\_train, y\_train, epochs=3, batch\_size=64)  
*# Final evaluation of the model*scores = model.evaluate(X\_test, y\_test, verbose=0)  
print(**"Accuracy: %.2f%%"** % (scores[1]\*100))

**Output**

****

**Question6**

Question 4 had accuracy of 88.49(Yaaayyy)

Question 5 had accuracy of 88.36(Naaayyy)

And On top of low accuracy, Question 5 took along time to run!

1. Datasets (if applicable)

Took Dataset from Kaggle and from Github according to requirements.

1. Parameters

No extra parameters except for those required in each question have been used. They do not require special mention as they have been explained in comments section of each code and in the workflow section

1. Evaluation & Discussion

The codes for all the questions are complete to the best of our knowledge and fulfil the criteria and guidelines mentioned by the tutor to fulfil while coding. Naming and coding conventions have been followed to the best of outrknowledge.

1. Conclusion

The resources are all our own and those which are downloaded are open source and are not liable to any type of copy right violation.  
The assignment and the lab problem is complete with Detailed codes and results of all 6 questions. All guidelines have been followed and we are now able to deduce and conclude that the main purpose of the complete assignment has been met.

**LAB Submission Guidelines (for both In Class and Online students):**

1. LAB submission is in a group of three students.

2. Submit your source code and documentation to GitHub and represent the work through wiki page properly (submit your screenshots as well. The screenshot should have both the code and the output)

3. Comment your code appropriately

4. Video Submission (2 – 3 min video showing the demo of the LAB, with brief voice over on the code explanation)

5. Submit **only** report at Turnitin in UMKC blackboard

6. Remember that similarity score should be less than **15%**

7. Use this link to submit your LAB#: <https://goo.gl/forms/l6q2rzkKCEGTigfp2>

8. Report should include below details

I. Introduction

II. Objectives

III. Approaches/Methods

IV. Workflow

V. Datasets (if applicable)

VI. Parameters

VII. Evaluation & Discussion

VIII. Conclusion