Machine Learning: Final Project Report

Team No: 24

Member: Abhijeet Kothari

TABLE OF CONTENTS

Abstract

❖ Projects:

I have chosen the below two projects as the final projects for Machine Learning:

- Type A project: Gotham City Cabs
    - Build a model to help Batman estimate the travel time of a cab from one point in the city to the other.

- Type B project: The Botanist
    - Develop a model to help botanist identify the type and decease of the plants.

❖ Motivation:

For Gotham City Cabs: I found the problem statement to be very interesting. More than often, we believe that time taken by a cab to travel from point A to point B is dependent upon the distance between these two points only. However, it is not always the case. There are other factors as well which significantly influence the travel time such as day of travel, time of travel, month etc. I was intrigued to understand how all these factors make an impact on the total duration of the trip. And, therefore, I decided to choose this project from Type A section.

For Botanist: I am fascinated by one of the features of Google photos which groups all the photos by persons' faces. I have always wondered how they do it. This used to feel like nothing short of some magic. This project gave me a chance to see how this magic works under the hood and introduced me to the various challenges/complexities involved in problems dealing with image classification.

❖ Project Outcomes:

For Gotham City Cabs: Developed a Linear Regression model which can predict the time taken by a cab to travel between two points with 0.22 Mean Squared Error on validation dataset. This model gave MSE of 318 seconds on the test dataset.

For Botanist: Developed an image classification model using PyTorch framework which can classify 38 different type of images and predict a label for an input image with 91% accuracy (this accuracy is on validation data set). But unfortunately, this model didn't perform in the same way on the test dataset.

Chapter: 1

Project: Gotham City Cabs (Type A)

❖ **Overview**:

1. Data:

This is a regression problem, where a model needs to be trained to predict the travel time of a cab from one point in the city to the other.
The input features of this problem are:

- pickup datetime: a variable containing a date and a time specifying the date and the time the taxi picked of a passenger.
- Number of Passengers: The number of passengers loaded to the cab.
- pickup x: This is a variable that represents the x coordinate of the location the taxi picked up the passenger.
- pickup y: This is a variable that represents the y coordinate of the location the taxi picked up the passenger.
- dropoff x: This is a variable that represents the x coordinate of the location the taxi dropped off the passenger.

	pickup_datetime	NumberOfPassengers	duration	pickup_x	pickup_y	dropoff_x	dropoff_y
0	2034-01-30 10:24:44	1	724	162.837930	341.187316	160.391473	367.907042
1	2034-03-09 23:10:11	1	127	150.375222	307.042187	152.623686	318.383231
2	2034-05-02 20:23:17	6	386	156.586093	333.063670	169.397955	315.001104
3	2034-06-21 17:51:55	1	1192	161.738726	344.609009	167.702052	310.817653
4	2034-05-15 18:38:15	2	315	174.419521	344.441542	165.780203	344.275954

- dropoff y: This is a variable that represents the y coordinate of the location the taxi dropped off the passenger.

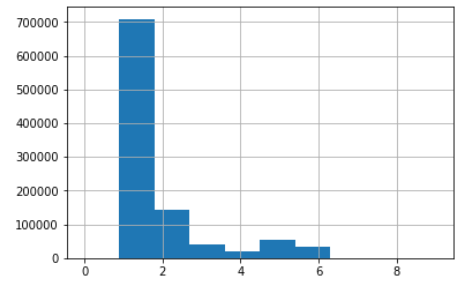
The response variable is:

- duration: which is the duration of the trip in seconds.

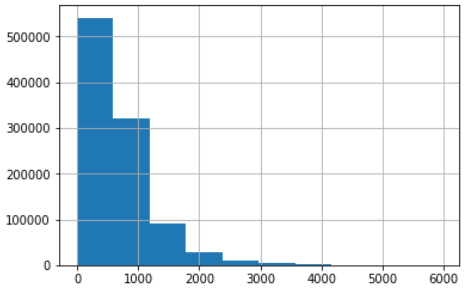
2. Exploratery Data Analysis

a. Co-relation Matrix for the original dataset:

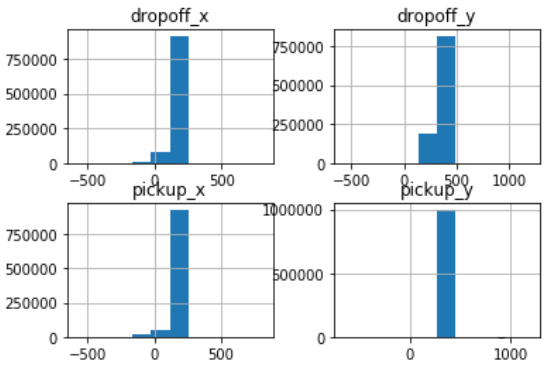
b. Histograms of Numerical Features:



(Distribution of passenger column)



(Distribution of duration column)



(Distribution of pickup and drop off columns)

Observations:

- Interesting point: rides with 0 passengers. Data is rightly skewed. Outliers are present.
- Rides with very short durations is present (including 0 seconds rides). Data is rightly skewed. Outliers are present.
- No significant co-relation found (from the heatmap)

❖ Feature Engineering & Data Cleaning

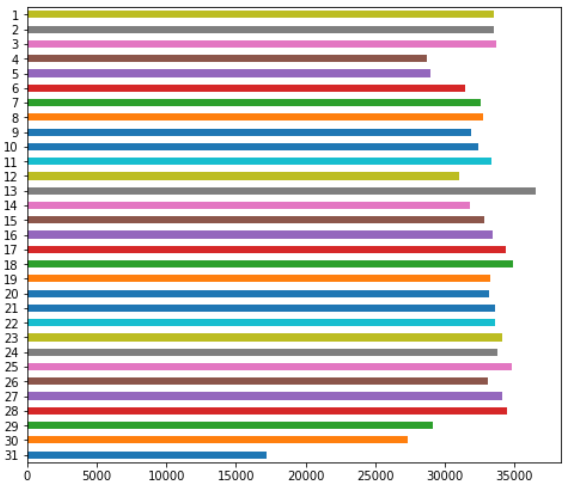
1. Feature Engineering:

New features had been created from the existing features such as:

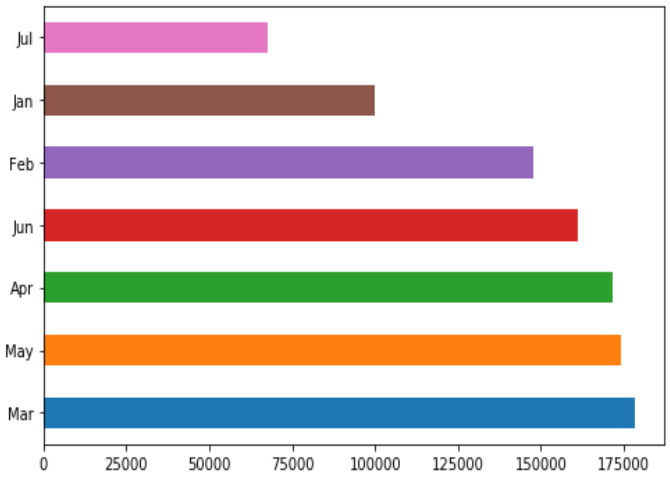
- Date
- Time
- Month
- Weekday
- Part of the day (morning, afternoon, etc.)
- Part of the month (early month, mid-month, etc.)
- Euclidean distance between the pickup point and drop off point.

2. EDA on derived columns

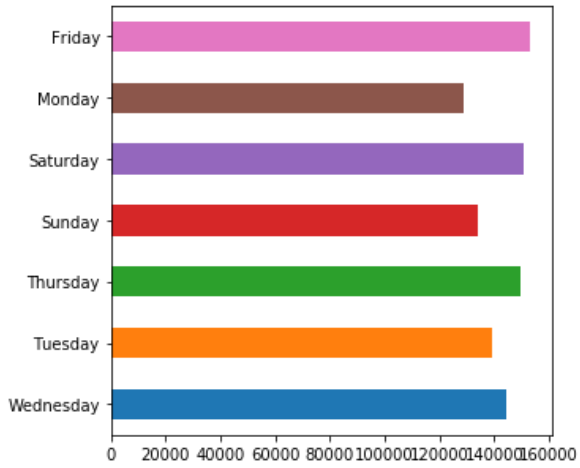
Bar chart of date column (date vs no. of rides)



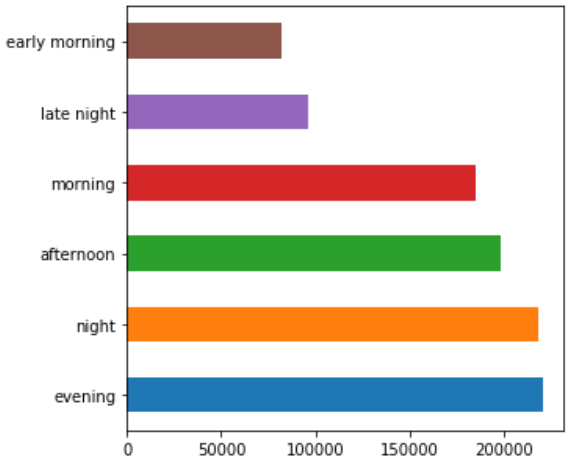
Bar chart of month column (month vs no. of rides)



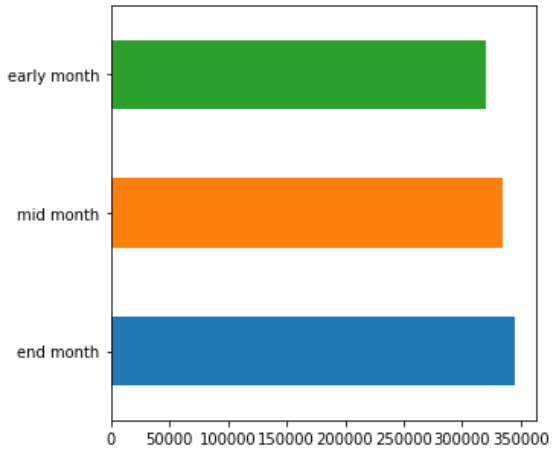
Bar chart of weekday column (weekday vs no. of rides)



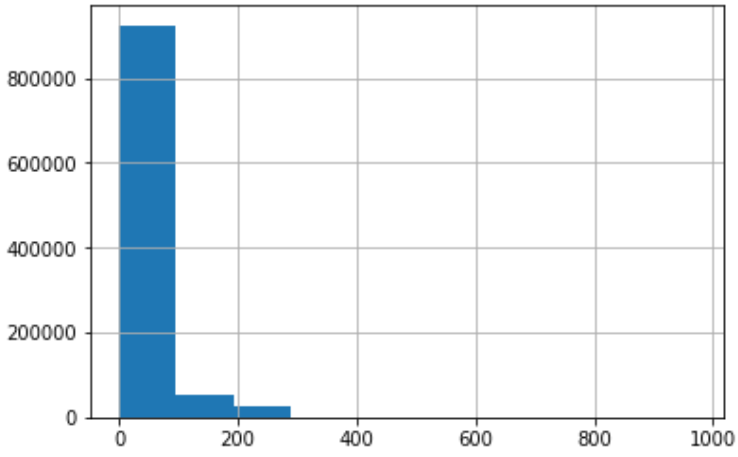
Bar chart of part of the day column



Bar chart of part of the month column against no. of rides



Histogram of Distance Column



Observations:

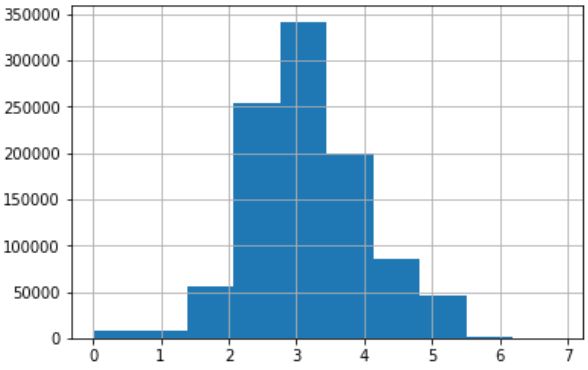
- Nearly equal number of trips on each day.
- Interesting point: Rides details are not present for August and later months.
- On Fridays, people took most number of trips. (Monday is 0)
- Afternoon, evening and night observed most number of rides. People took less rides in early morning and late night.
- No one part of the month witnessed significant higher rides than others.
- Data is rightly skewed for distance column. Outliers are present.

3. Data Cleaning:

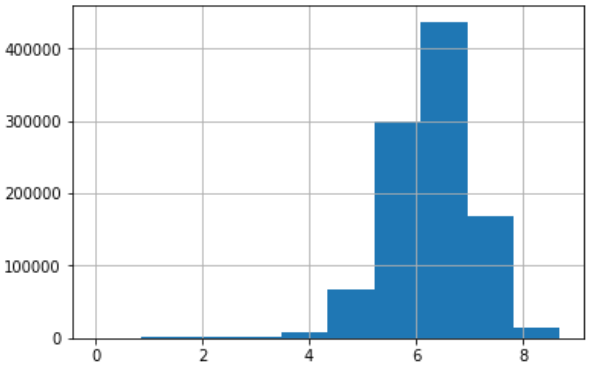
For cleaning the data, following operations were performed:

- Dropped some of the columns for instance pickup_datetime, year. time, pickup_x, pickup_y, dropoff_x, dropoff_y, date.
- Removed rides with 0 riders.
- Handled outliers and skewed data with the help of log transformation
- Encoded categorical features.

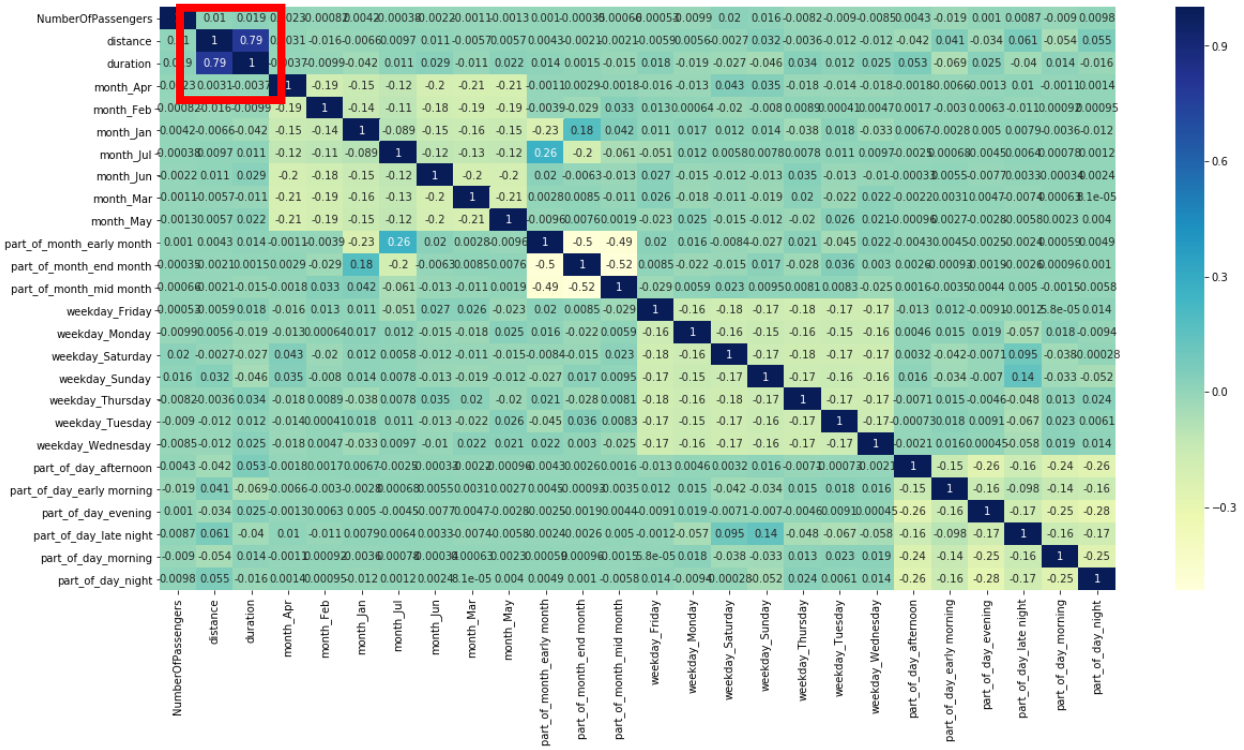
4. EDA after cleaning:



Histogram of Duration Column after Removing Skewness



Histogram of Distance Column after Removing Skewness



Observations:

- Histograms for duration and distance features are in better shape now. They are skewed as they were before.
- From the heat map it can be seen that there is a high co-relation between distance and duration.

❖ Problem Analysis

1. Models tested:

Following models were trained and tested:

- Linear Regression Model:
- Ridge Model
- Lasso Model
- Random Forest Regressor Model
- Decision Tree Regressor

2. Metrics used to compare models:

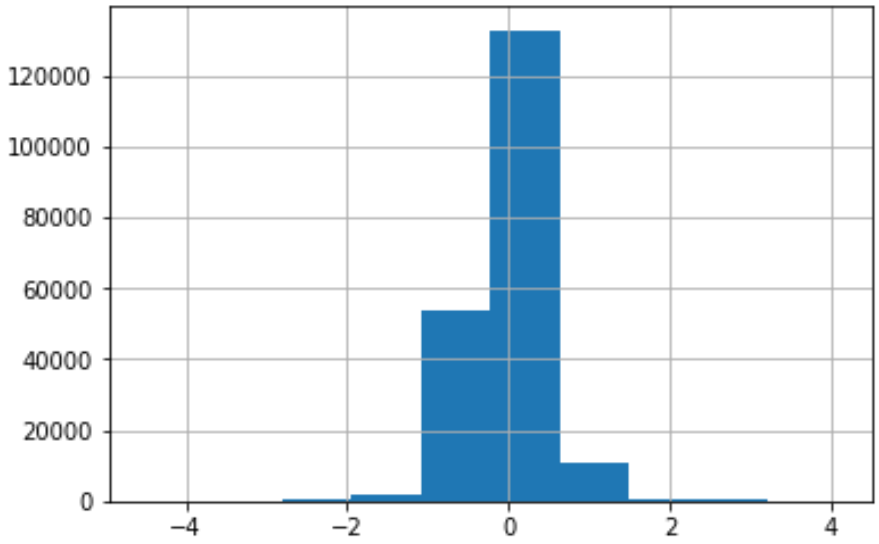
- R2 score (R-Square Score)
- MSE (Mean Squared Error)
- MAE (Mean Absolute Error)

3. Results:

Below table lists efficiency matrices for the above-mentioned model. Linear regression model performed better than other models.

Model	R2 Score	MSE	MAE
Linear Regression Model	0.660	0.202	0.300
Ridge Model	0.654	0.210	0.316
Lasso Model	0.655	0.211	0.357
Random Forest Regressor Model	0.621	0.225	0.459
Decision Tree Regressor	0.388	0.366	0.444

Histogram for the residual error of the Linear Regression Model was also checked.



Observation: The histogram of residual error for Linear Regression Model doesn't seem skewed and it resembles the shape of bell curve a bit as well. So this model has been chosen for making the predictions.

Exponents values were calculated of the predictions to obtain the actual predicted duration. (it is because log transformation was performed on the duration feature before the training to handle the skewness)

❖ Conclusion:

This was a regression problem where there was a scope of data cleaning and of deriving new features from existing features. In total, five models were compared in terms of R2 Score, MSE, and MAE and it is found that Linear Regression Method performed better than others. R2 Score, MSE and MAE score for the Linear Regression model stood at **0.660**, **0.202**, and **0.300** respectively for validation dataset. When the prediction was made for the competition test data file, this model produced the **MSE value of 318 seconds**.

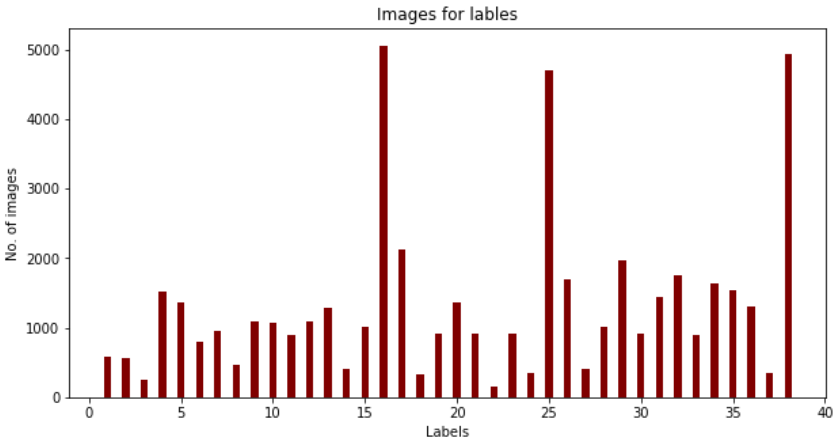
Alfred, who needs help to come up with a good prediction of the taxi trip duration between multiple points of the Gotham city, can make use of the predictions made by Linear Regression Model. It can significantly help with Batman's missions.

Chapter 2

Project: Botanist (Type B)

❖ Overview:

This is a classification problem, where the goal is to train a model which is fed with an image of a leaf, and predicts a label corresponding to the type and decease of the plant.



To handle this machine leaning problem, concept of Transfer learning has been utilized. A pre-trained model has been used. Pre-trained models are Neural Network models trained on large benchmark datasets like ImageNet. Other researchers and practitioners can use these state-of-the-art models instead of re-inventing everything from scratch. The word pre-trained here means that the deep learning architectures have been already trained on huge dataset and thus carry the resultant weights and biases with them.

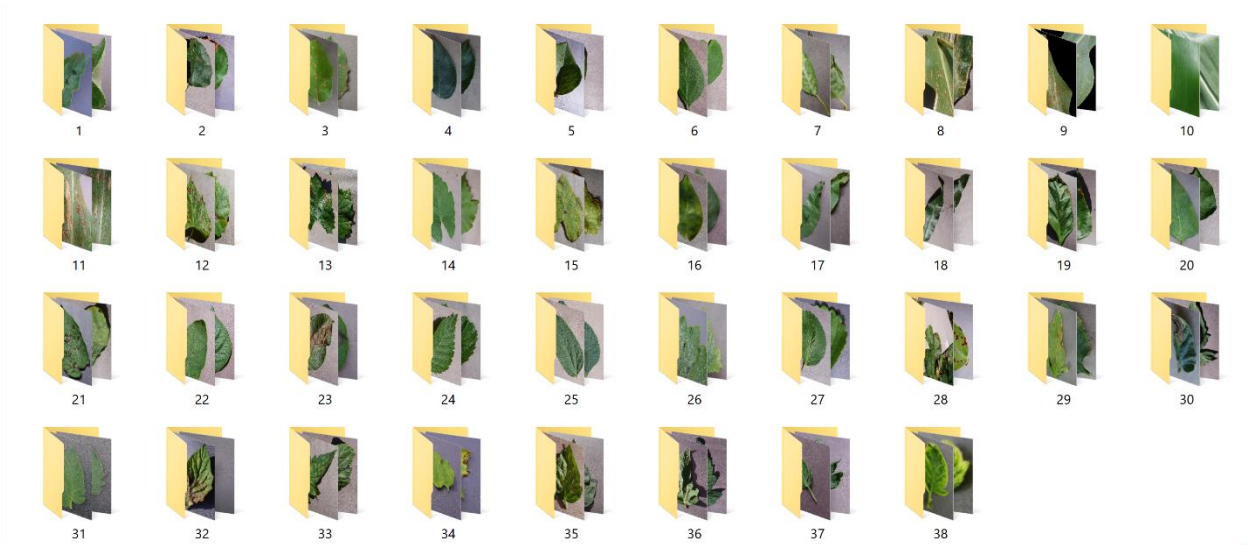
The entire process consists of the following main steps:

- Reading the input image
- Performing transformations on the image. For example – resize, center crop, normalization, etc.
- Forward Pass: Use the pre-trained weights to find out the output vector. Each element in this output vector describes the confidence with which the model predicts the input image to belong to a particular class.

- Based on the scores obtained (elements of the output vector we mentioned in step-3), display the predictions.

❖ Data Transformation Steps:

Firstly, the images were segregated and were put in their respective label's folder. The below image depicts the output of this operation:



Later, after importing the images in a notebook, following steps were performed on the images (in order to generalize the network for better performance):

- Random Rotation
- Random Scaling
- Cropping (224 X 224 pixels)
- Color Channel Normalization (Shifted each color channel to be centered at 0 and range from -1 to 1)

The above transformations are performed on the training Dataset. For validation dataset and testing dataset, only resize and crop operations are performed.

❖ Problem Analysis

Models Tested:

Following pre-trained model has been tested for this given problem:

- VGG
- ResNet
- DenseNet

Steps Performed:

Defined a new untrained feed-forward network as a classifier with following specification:

- Input layers: 1024
- Hidden layers: 256
- Output layers: 38
- Activation Function: ReLU
- Loss Function: Negative Loss Likelihood Function

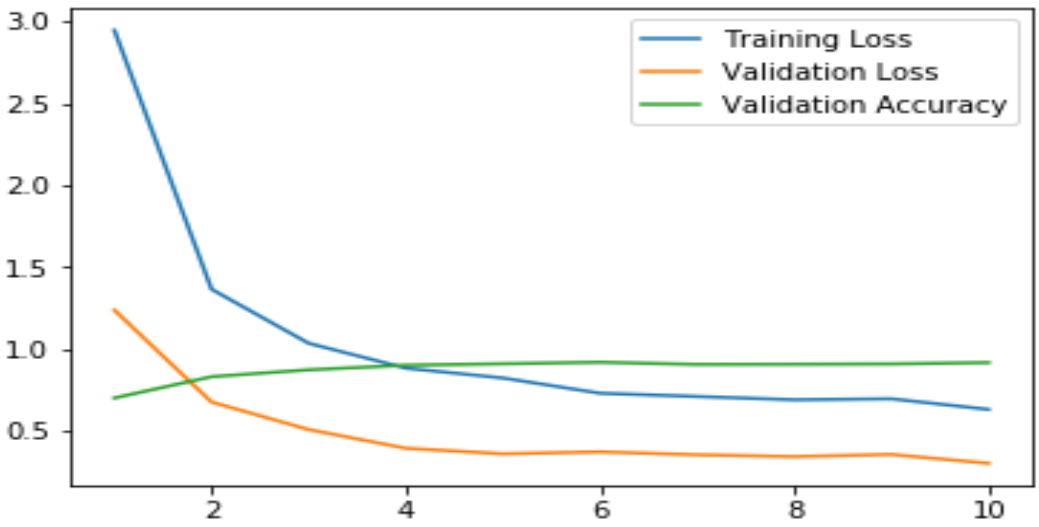
Changed the classifier of pre-trained model with our feed-forward classifier. Train the classifier layers using backpropagation using the pre-trained network to get the features. Following steps were performed:

- Iterate through data loader.
- Calculate the classifier parameters through the 'model.forward()' function.
- Calculate loss.
- Modified the parameters of classifier using back propagation.
- Calculated training loss, validation loss and validation accuracy for each epoch.

Results:

I observed that DenseNet mode performed better than VGG and ResNet models. For the DenseNet model, following results were obtained.

```
Epoch: 1/10..  Training Loss: 2.945..  Validation Loss: 1.236..  Validation Accuracy: 0.700
Epoch: 2/10..  Training Loss: 1.364..  Validation Loss: 0.675..  Validation Accuracy: 0.830
Epoch: 3/10..  Training Loss: 1.034..  Validation Loss: 0.507..  Validation Accuracy: 0.872
Epoch: 4/10..  Training Loss: 0.882..  Validation Loss: 0.393..  Validation Accuracy: 0.900
Epoch: 5/10..  Training Loss: 0.822..  Validation Loss: 0.359..  Validation Accuracy: 0.909
Epoch: 6/10..  Training Loss: 0.729..  Validation Loss: 0.371..  Validation Accuracy: 0.918
Epoch: 7/10..  Training Loss: 0.709..  Validation Loss: 0.353..  Validation Accuracy: 0.905
Epoch: 8/10..  Training Loss: 0.689..  Validation Loss: 0.342..  Validation Accuracy: 0.906
Epoch: 9/10..  Training Loss: 0.695..  Validation Loss: 0.355..  Validation Accuracy: 0.908
Epoch: 10/10..  Training Loss: 0.630..  Validation Loss: 0.301..  Validation Accuracy: 0.916
```



Predictions for test file:

To find the labels of test images with the selected model, following steps were performed.

- Iterated through images on test dataset one by one and transformed each image (resize, crop, color channel normalization).
- Converted the image to torch float tensor.
- Passed the tensor to the model and obtained the top predicted label.
- Save the image name and it's predicted label in a dictionary.
- Created the data frame from the dictionary and reindexed it accord to the test response file.

❖ Conclusion

This was an image classification problem that utilized PyTorch framework. Different types of pre-trained models were tested such as VGG, ResNet and DenseNet. Out of these model, DenseNet121 model provided much better accuracy than others. The accuracy stood at 91.6% for this model. Although, this model didn't perform in the same way on test dataset and produced a low accuracy score. However, It brings the scope of trying other Deep Learning Models on dataset such as CNN to improve the accuracy of the model.