Use of Location Data for effective taxi business improvement in New York

Rohith Kumar Punithavel

I. Introduction

New York City, being the most populous city in the United States, has a vast and complex transportation system, including one of the largest subway systems in the world and a large fleet of more than 13,000 yellow and green taxis, that have become iconic subjects in photographs and movies.

The subway system digests the lion share of NYC's public transport use, but the 54% of NYC's residents that don't own a car and therefore rely on public transportation still take almost 200 million taxi trips per year!

New york city is a dream destination for many people around the world. This place is densely populated and prone to heavy traffic all throughout the day. By analysing the mode of transportation in this part of the world, it is understood that it is mostly public transport and taxis owing to heavy traffic and high parking prices. This project aims to help taxi drivers in finding passengers all throughout the day such that their profits are maximised and also help taxi organizations to increase their annual profits and shares.

There are many problems faced by taxi company some of them are:
1. Proper pickup location for passengers based on timing and day: for example during the afternoon walmarts, costco and safeway are the place to pickup people whereas on friday nights downtown's are the trending place.
2. Assigning the cabs to passengers efficiently and also determine the duration of the current trip so it can predict when the cab will be free for the future trip.

These two problems are solved based on pickup/dropoff locations

The Beneficiaries are Cab drivers and taxi companies and the target audience is the public of New York.

II. Dataset

The dataset used is https://www.kaggle.com/c/nyc-taxi-trip-duration

The dimensions of the dataset is 1458644 X 11.

The description of the dataset from the source is:

1. id - a unique identifier for each trip
2. vendor_id - a code indicating the provider associated with the trip record
3. pickup_datetime - date and time when the meter was engaged
4. dropoff_datetime - date and time when the meter was disengaged
5. passenger_count - the number of passengers in the vehicle (driver entered value)
6. pickup_longitude - the longitude where the meter was engaged
7. pickup_latitude - the latitude where the meter was engaged
8. dropoff_longitude - the longitude where the meter was disengaged
9. dropoff_latitude - the latitude where the meter was disengaged
10. store_and_fwd_flag - This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server - Y=store and forward; N=not a store and forward trip
11. trip_duration - duration of the trip in seconds

Additional API
Foursquare Location API - https://developer.foursquare.com/ - for location information

How Dataset is used:

The dataset is used to primarily identify pickup locations based on timing and day and to cluster places based on these two categories as well. The pickup locations are used to cluster and trip duration is used to estimate the time left so that they can schedule the pickup for the next customer. The longitude and latitude is used for visualization.

III. Preprocessing

A new column for the day is added based on pickup_datetime and dropoff_datetime using pd.series. After that with time four new categories of data is processed namely, Morning (4:01am - 11:00am), Midday(11:01 am - 16:00pm), Evening(16:01pm -

22:00pm) and latenight(22:01pm - 4:00am). Based on these four categories the data is grouped. My initial plan of execution is based on location name, but to retrieve location name we need external location API called Foursquare API which allows only 900 free calls per day so the visualization is done for small dataset and displayed but not trained because there is too much overfitting to data ie small amount of data only used. The outliers for trip duration are removed and values are changed to the median of the distribution. Then the distribution still continued to be skewed, hence was split based on trip duration. The trips with duration 1 are removed off the table. The records pertaining to passenger count less than 6 per vendor id is also removed. The distance is then calculated, and the trips with zero distance are converted to mean of the distribution and since distances are of variable scale all are converted to log scale and values greater than 2 are removed. The speed is also calculated.

IV. Methodology

There are two models implemented:

1. Linear Regression;

Linear Regression models are used to estimate values based on the data points on which it is trained. Here the target is trip_duration based on which fares depend. The residuals are used for plot as variations in features are different.

2. KMeans

KMeans is a clustering technique which combines points that are closer together. The one-hot encoded values for selected features are used to train and clustered. Here clusters =4 hence they should be clustered based on whether they are picked up in morning, midday, evening and latenight and hence plotted.

V. Results

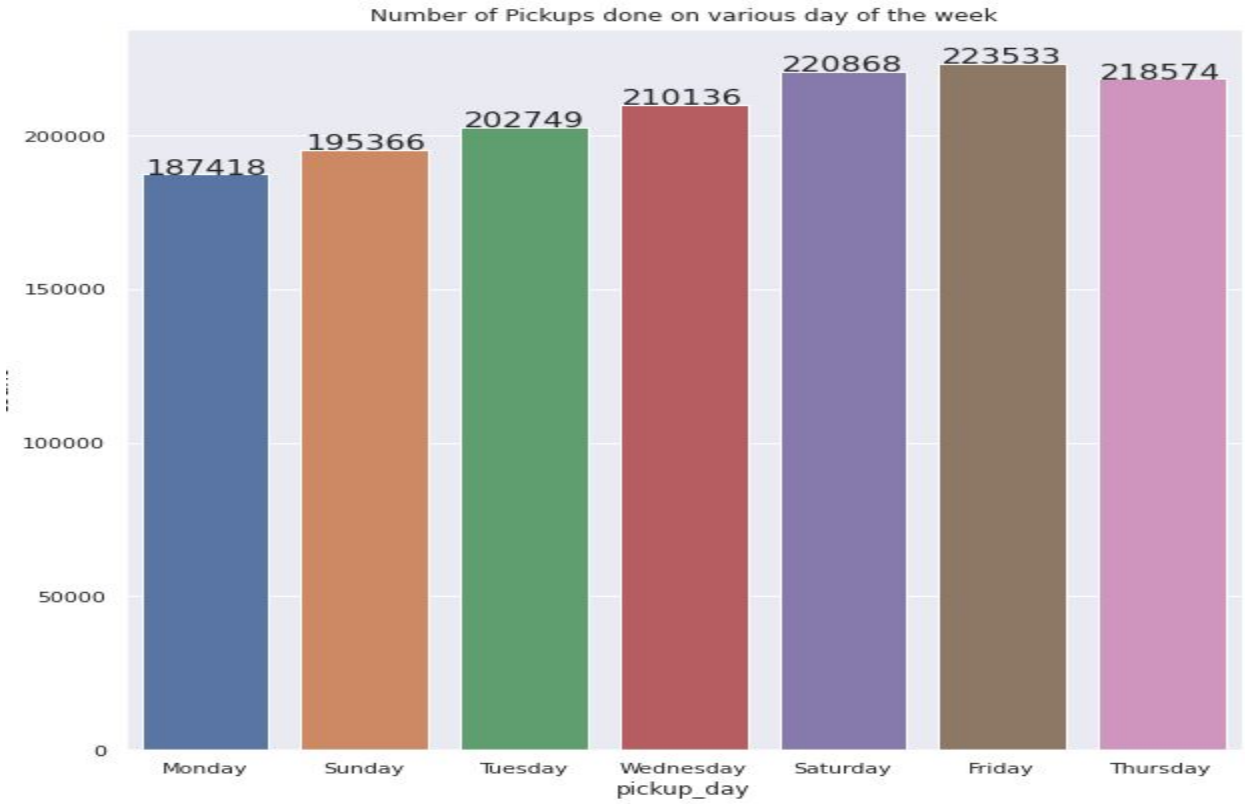

Fig.1 Number of pickups done on various day of the week

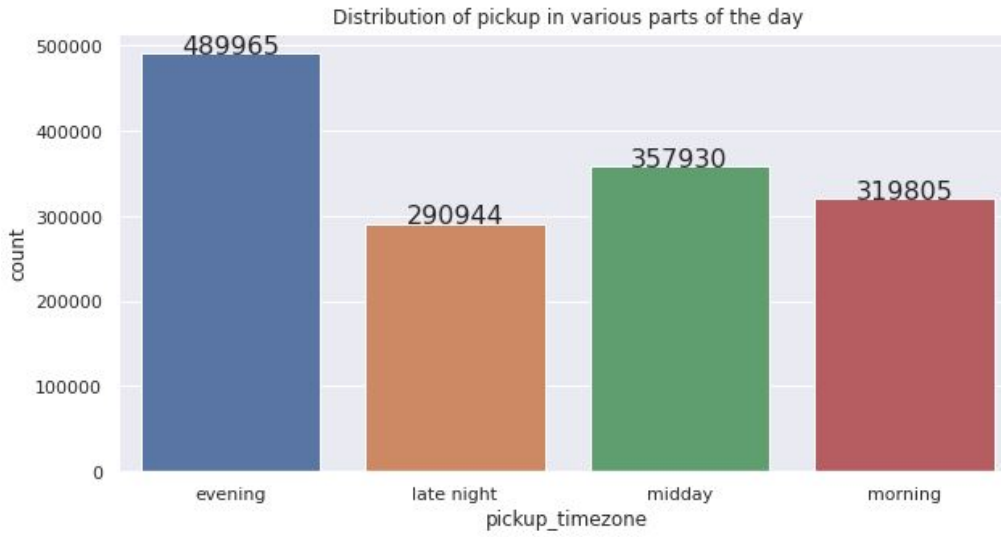

Fig.2 Distribution of pickup in various parts of the day

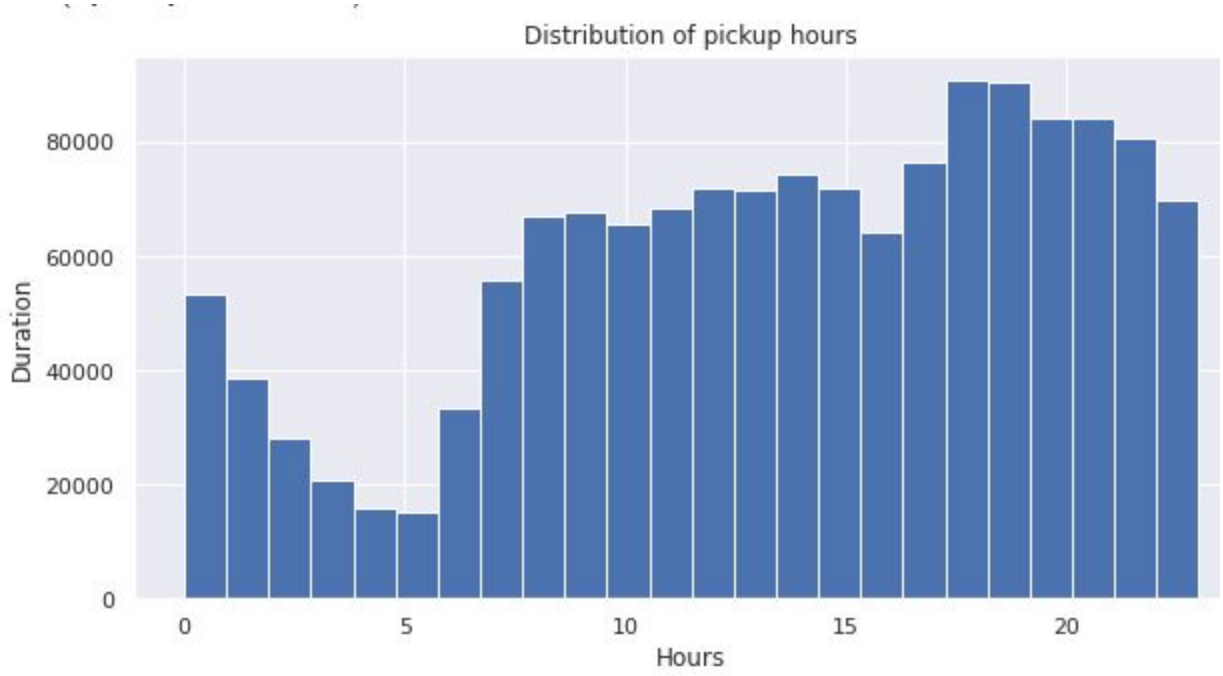

Fig.3 Distribution of pickup hours

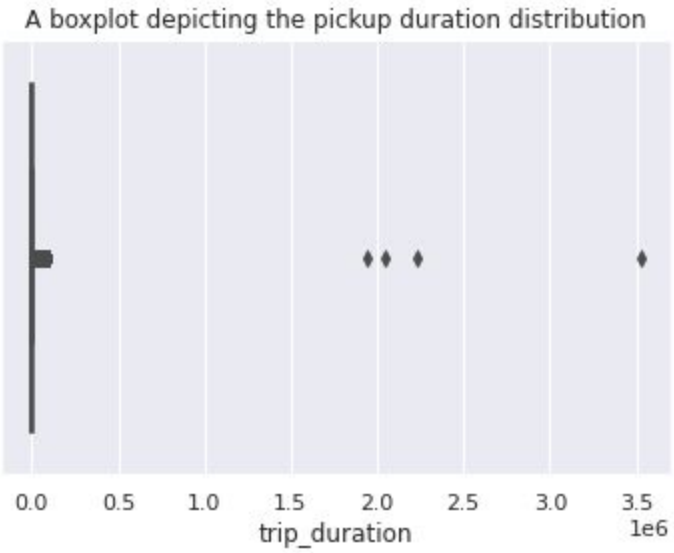

Fig.4 Boxplot to detect outliers

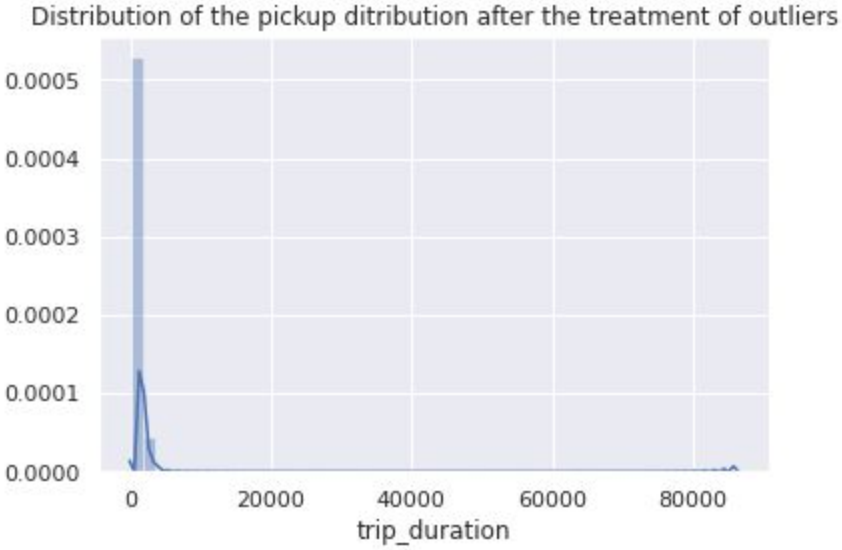

Fig.5 Plot to check distribution trip duration after treating outliers

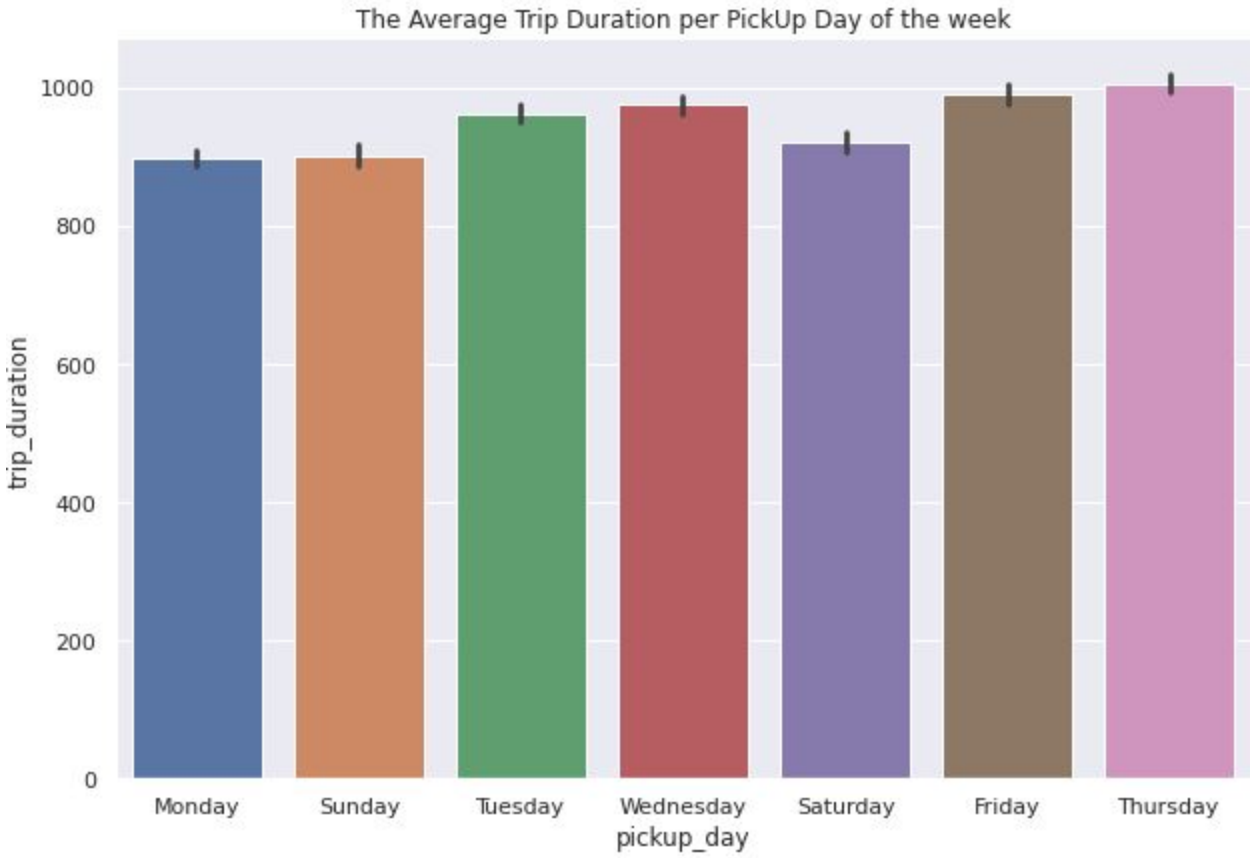

Fig.6 Average trip duration per pickup day of the week

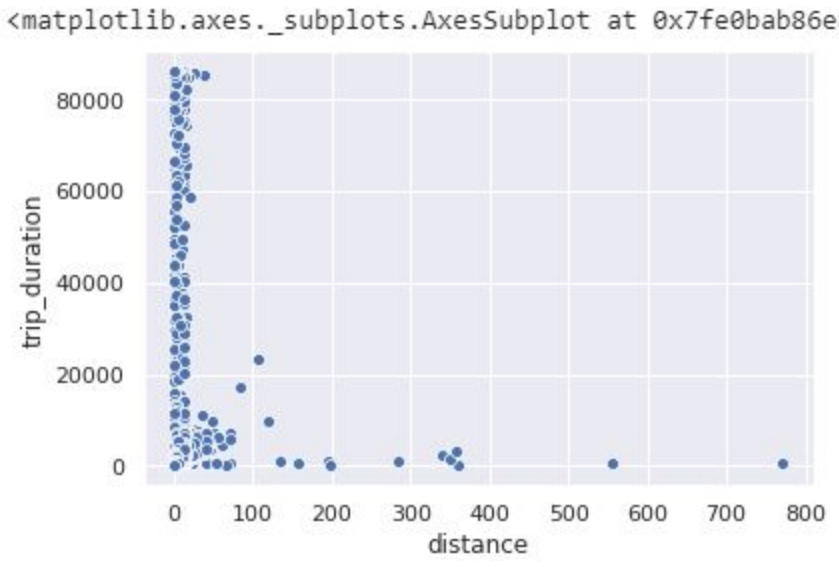

Fig.7 trip duration vs distance plot

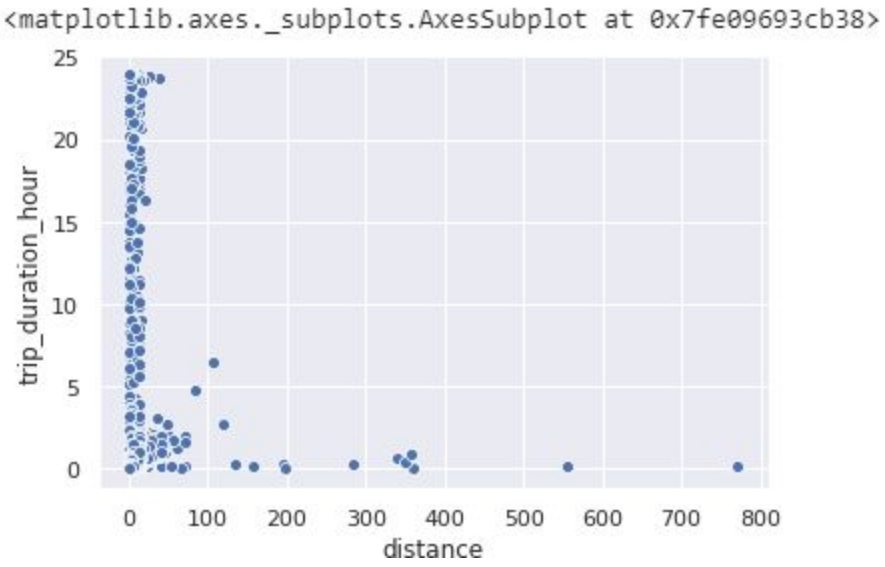

Fig.8 trip duration hour vs distance plot

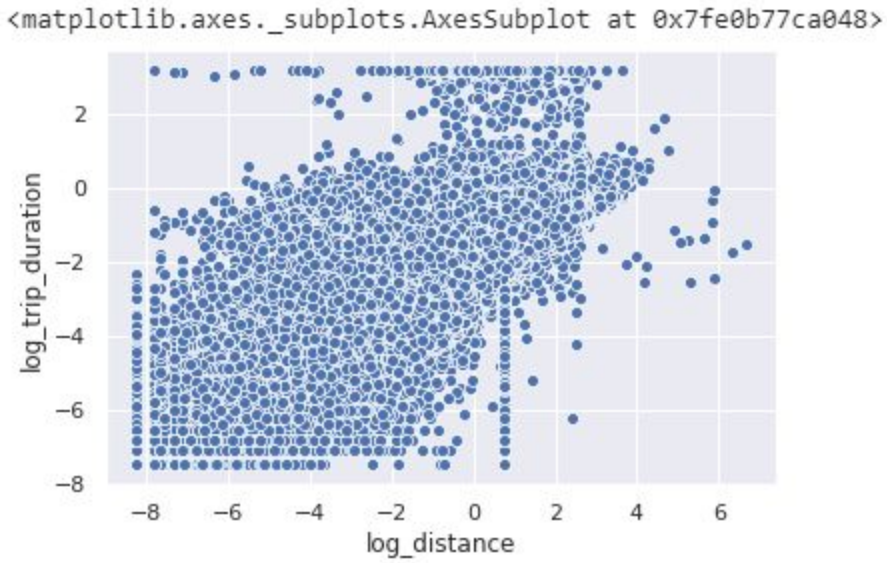

Fig.9 log trip duration vs log distance

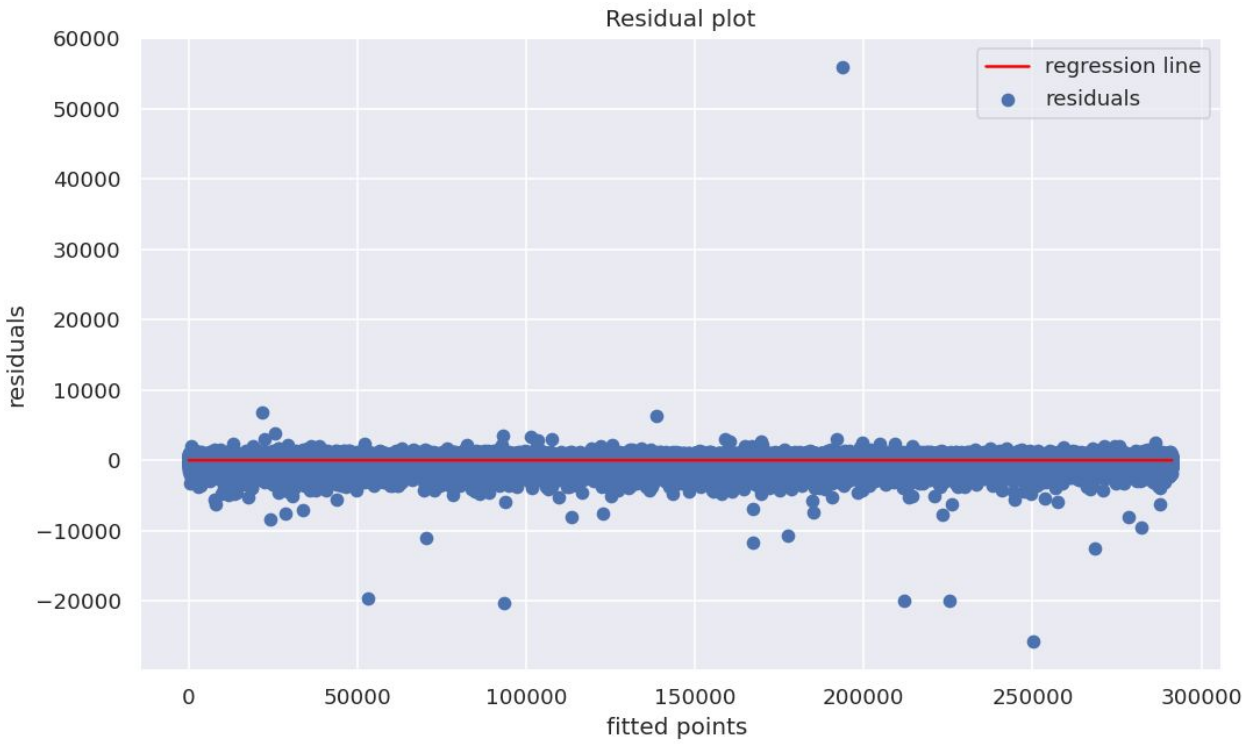

Fig.10 Linear Regression plot

```
[40] y_pred=modelfit(alg1, X_train, X_test,base_line_col,target_col)
     coef1 = alg1.coef_
     print('The coeffient is {}'.format(coef1))
```

```
The baseline model

Model Report
RMSE on Train Data: 485.7
CV Score : Mean - 482.8 | Std - 77.21 | Min - 432.5 | Max - 729.7
RMSE on Test Data: 453.3
The coeffient is [[170.25861852]]
```

```
alg2 = LinearRegression(normalize=True)
y_pred=modelfit(alg2, X_train, X_test, predictor_cols,target_col)
coef1 = pd.Series(alg2.coef_[0], predictor_cols).sort_values()
coef1.plot(kind='bar', title='Model Coefficients')
```

```
Model Report
RMSE on Train Data: 473
CV Score : Mean - 469.8 | Std - 79.05 | Min - 418.7 | Max - 722.1
RMSE on Test Data: 440
<matplotlib.axes._subplots.AxesSubplot at 0x7fe0bb9a1860>
```

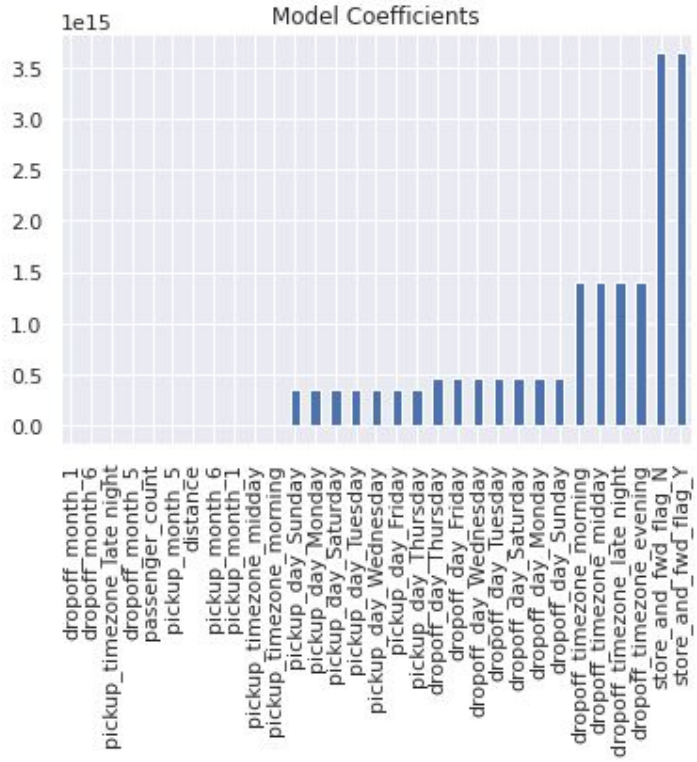

Fig.11 RMSE of Baseline and predictor model

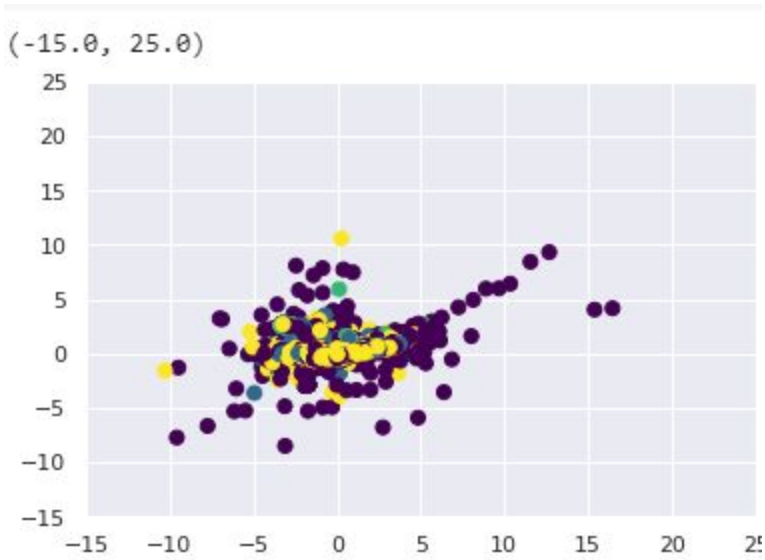

Fig.12 Kmeans plot

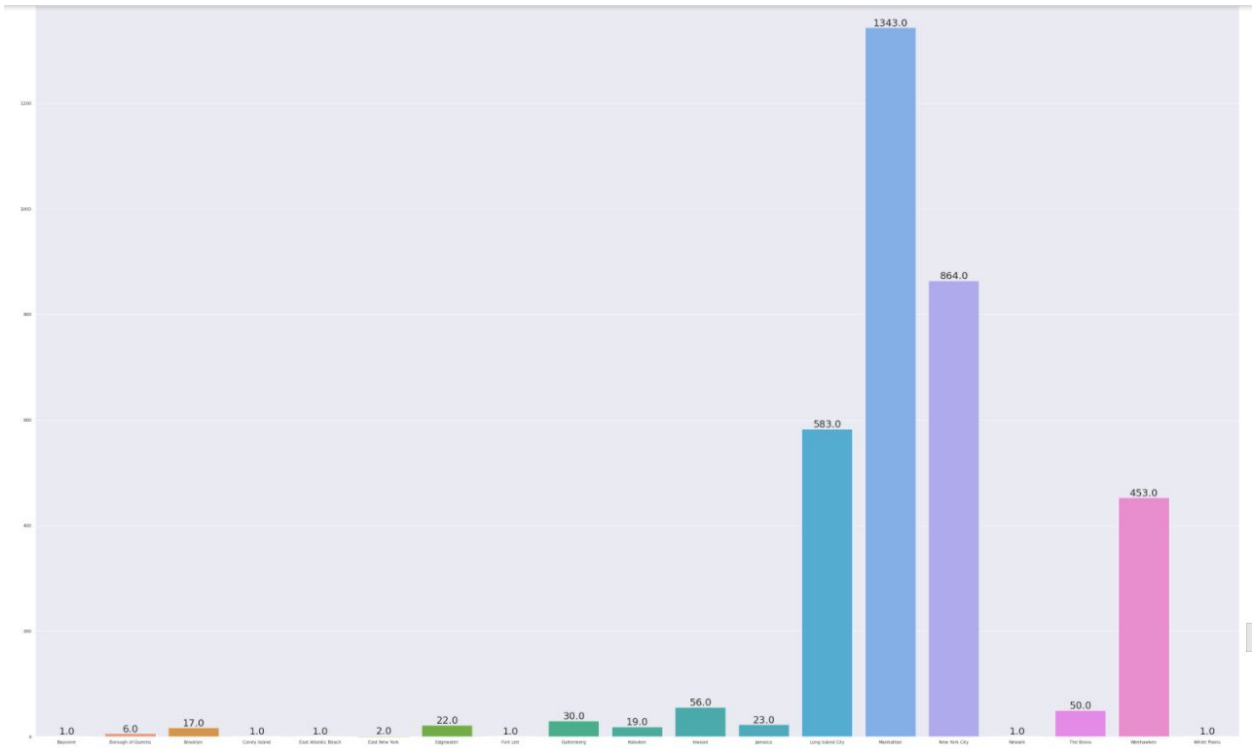

Fig.13 Morning Distribution Vs Number of trips

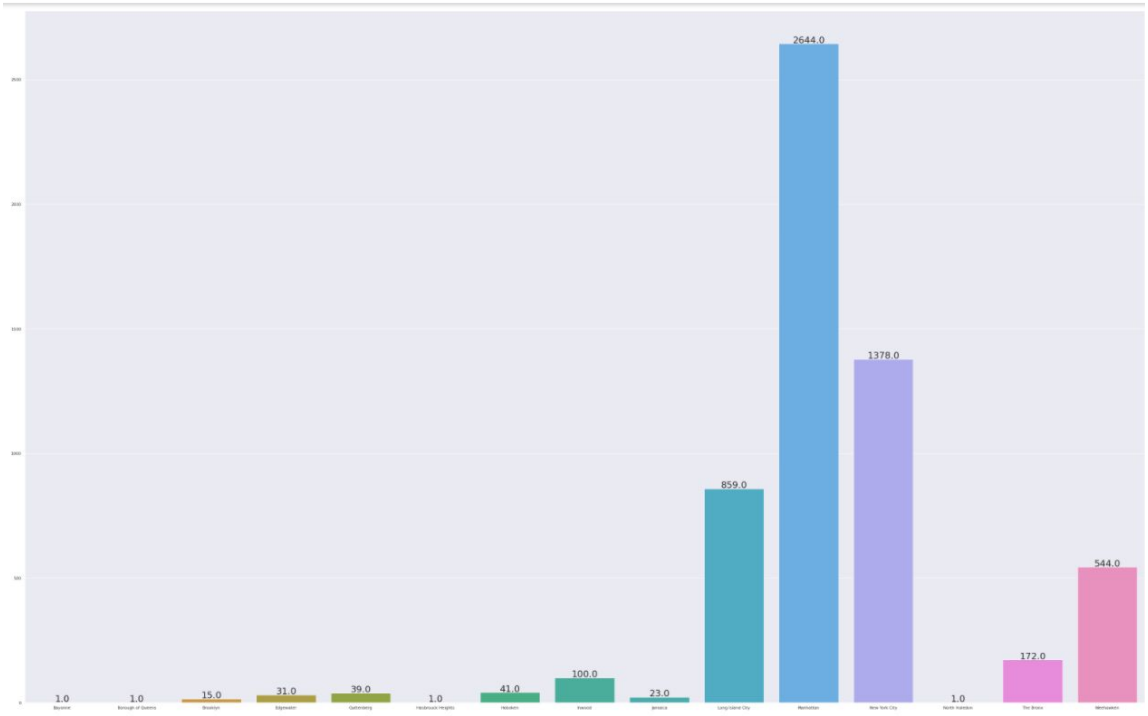

Fig.14 Midday Distribution Vs Number of trips

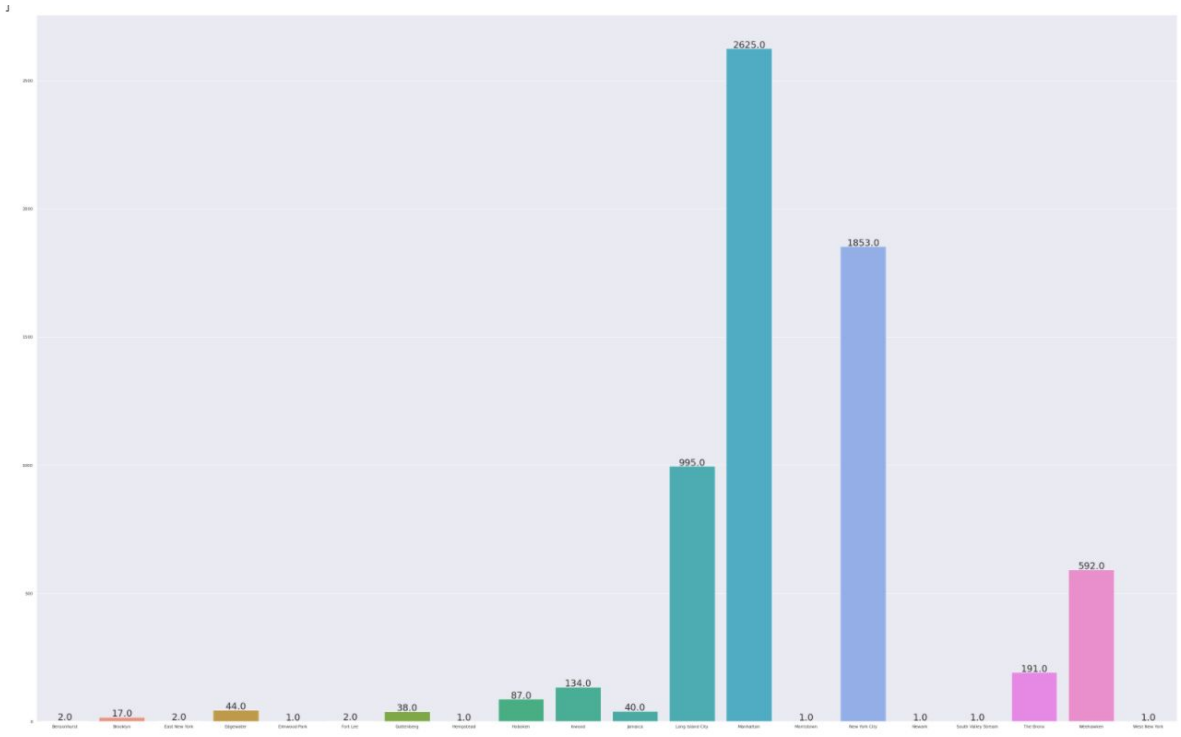

Fig.15 Evening Distribution Vs Number of trips

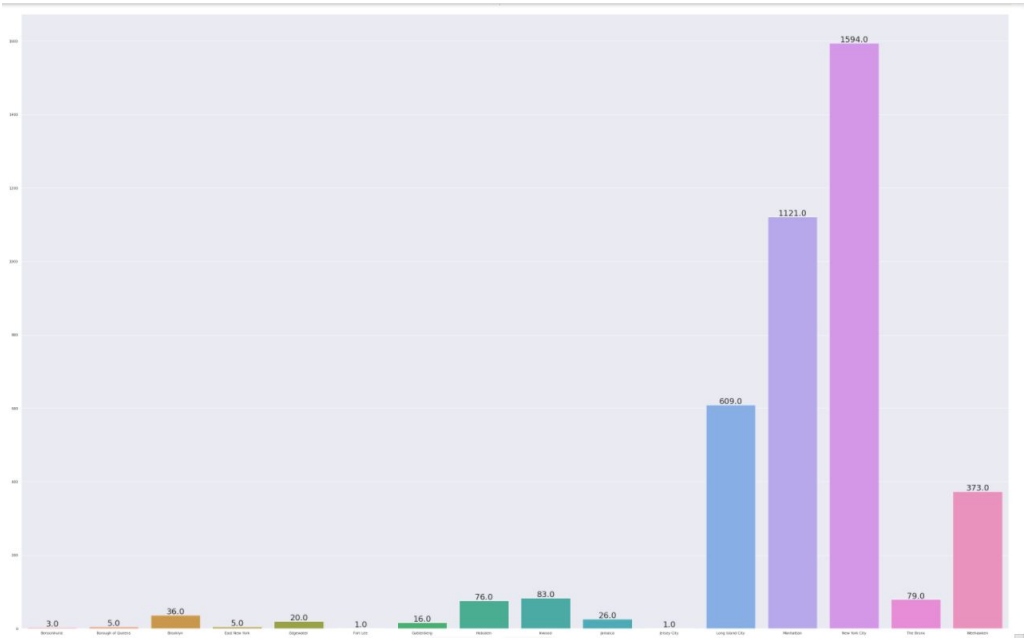

Fig.16 LateNight Distribution Vs Number of trips

VI. Discussion

In this project, the Linear Regression model fitted really well, thus we can say that we can accurately estimate the trip duration and hence the fare. The fare estimation needs some more data on fares and hence can be extended to the future. As far as KMeans is concerned, the clustering did happen but didn't go accurately as expected due to absence of location name and hence did not have that extra information to accurately cluster and also requires a really good GPU to perform a lot of computations. This requires a lot of work to be done to the dataset for my specific problem.

VII. Conclusion

Thus this project has been a really good learning experience for me and requires a lot of modifications to the dataset to work as per my requirement. From Fig.13 to Fig.16 , denotes the pickup location during various parts of the day to the driver and hence boosts their profits. The Regression model also helps in effectively assigning customers to taxis during peak hours, but the downside is the passenger with a short trip needs to wait a few minutes longer when the passenger with a long trip books a taxi at the same time. This model can further be improved by adding few more features and hence predicts output accurately.