EDA Capstone

Capstone Project

NYC Taxi Trip Time Prediction

Vishakha Kumari
Shivam Kumar
Abhinav Akotkar
Durgesh Shukla

Content

- Introduction
- Data Description and cleaning
- New FeaturesCreating
- Univariate Analysis
- Bivariate Analysis
- Location Visualization On Map
- Correlation Heatmap
- Linear Regression
- Lasso Regression
- XGBoost Model
- LightGBM Model
- Model Summary
- Conclusion

Introduction

- Our task is to build a model that predicts the total ride duration of taxi trips in New York City.
- Primary dataset contains NYC Taxi and Limousine Commission, which includes pickup time, geo-coordinates, number of passengers, and several other variables.

- The dataset is based on the 2016 NYC Yellow Cab trip record data made available in Big Query on Google Cloud Platform. The data was originally published by the NYC Taxi and Limousine Commission (TLC). The data was sampled and cleaned for the purposes of this project. Based on individual trip attributes, you should predict the duration of each trip in the test set.

Data Description and cleaning

SHAPE OF DATA

Following is the shape of data
Rows: 1458644
Columns: 11
Null values - 0 (There are no NaN or null values in our dataset)

Data contains the training set (contains 1458644 trip records)

Data Description and cleaning- Column description

- Id - a unique identifier for each trip
- vendor_id - a code indicating the provider associated with the trip record
- pickup_datetime - date and time when the meter was engaged
- dropoff_datetime - date and time when the meter was disengaged
- passenger_count - the number of passengers in the vehicle (driver entered value)
- pickup_longitude - the longitude where the meter was engaged
- pickup_latitude - the latitude where the meter was engaged
- dropoff_longitude - the longitude where the meter was disengaged
- dropoff_latitude - the latitude where the meter was disengaged
- 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
- trip_duration - duration of the trip in seconds

New FeaturesCreating

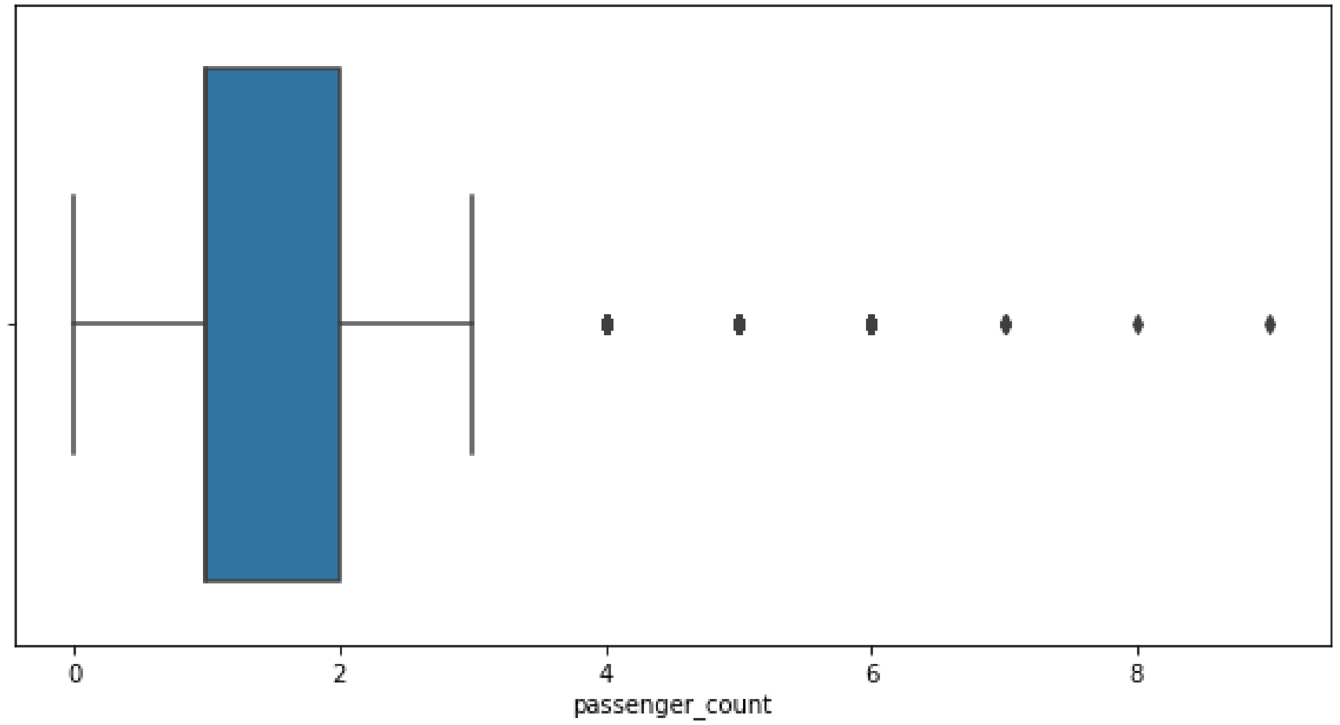
Here we added some new features for our ease of processing.

- pickup_weekday: This will contain the day on which ride was taken.
- dropoff_weekday: This will contain the day on which ride was ended.
- pickup_hour: time at which ride was started.
- dropoff_hour: time at which ride was stoped.
- month: month of the ride
- pickup_weekday_num: number of the day on which ride was taken, taking Monday as 0.

Univariate Analysis

Number of Passengers Per Cab

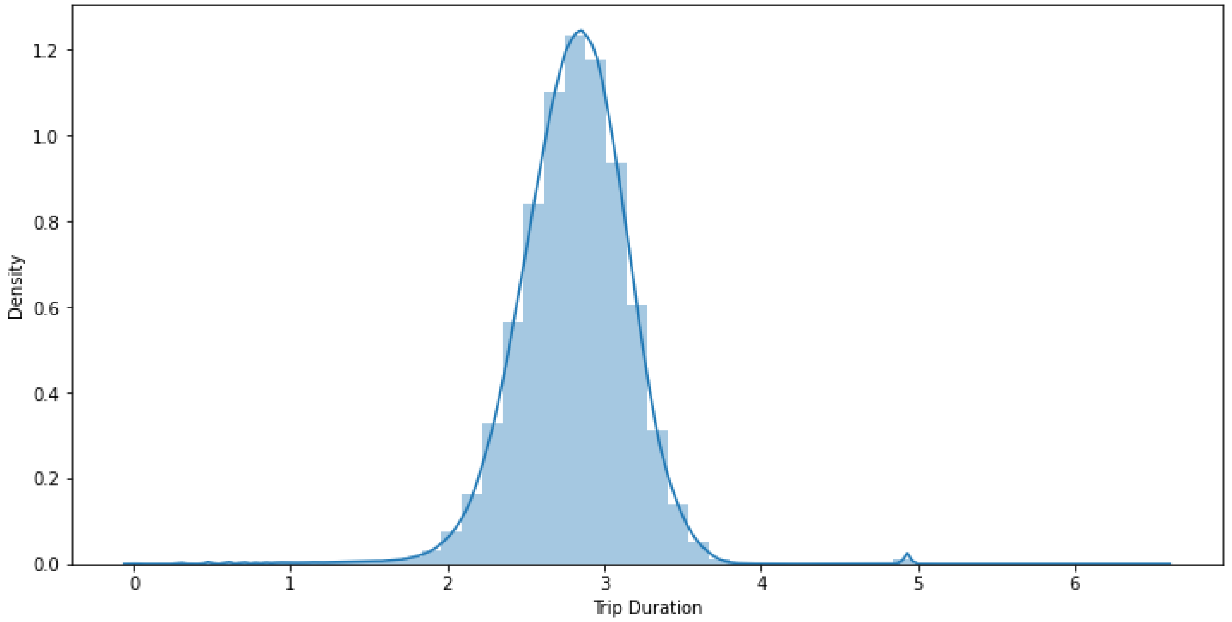
to check whether there are any outliers in our dataset



From the above Box Plot we can clearly notice there are some outliers in our dataset. It shows number of passengers per taxi is more than 7.

In most of the trips passenger number is between 1 or 2

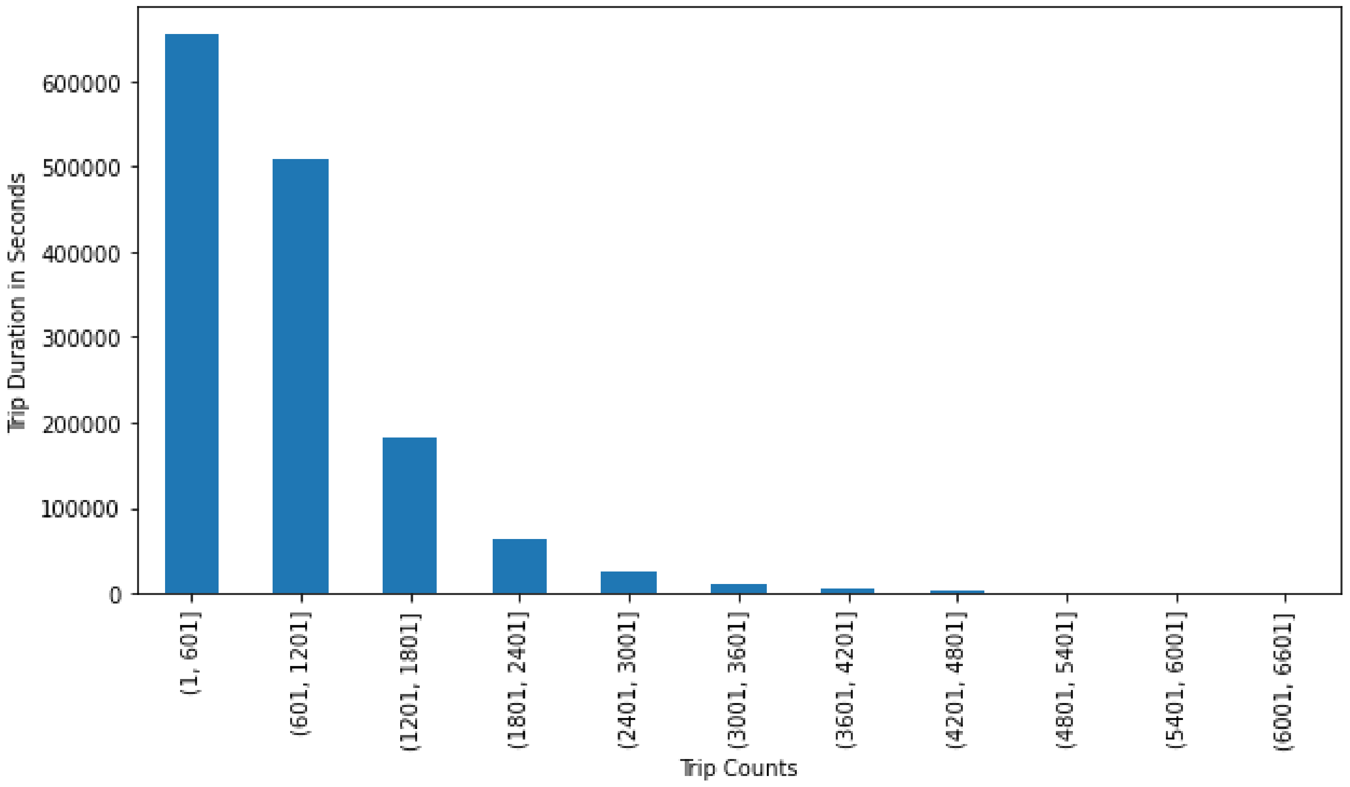
Trip Duration



We can spot an outlier near 5 in the above plot.

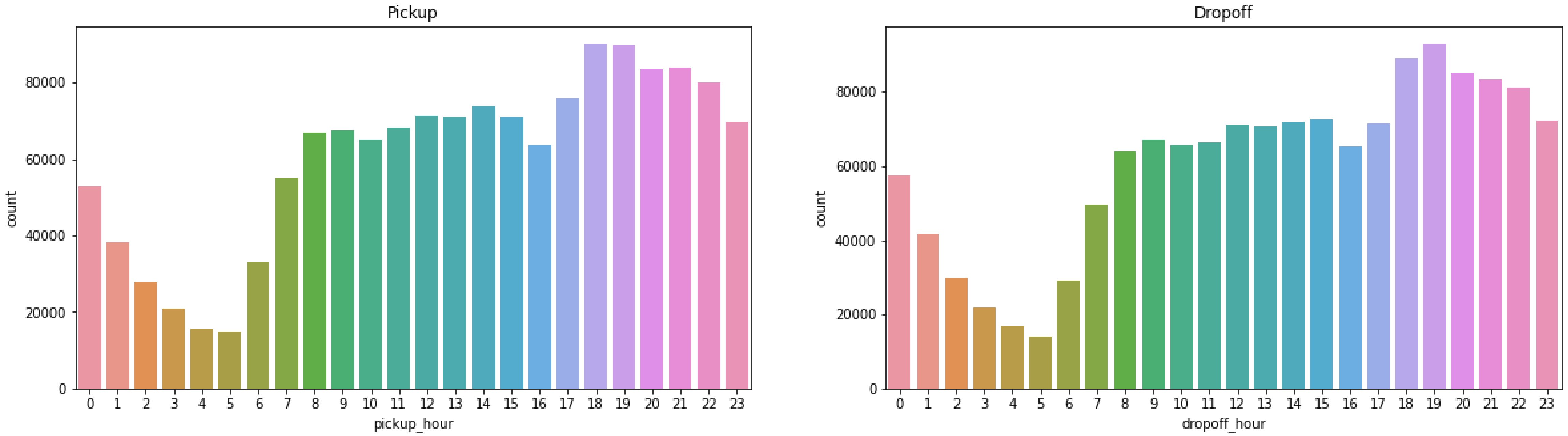
Univariate Analysis

Number of Trips Taken per Minute



Most of the trips took around 1800 seconds

Total Trips per Hour

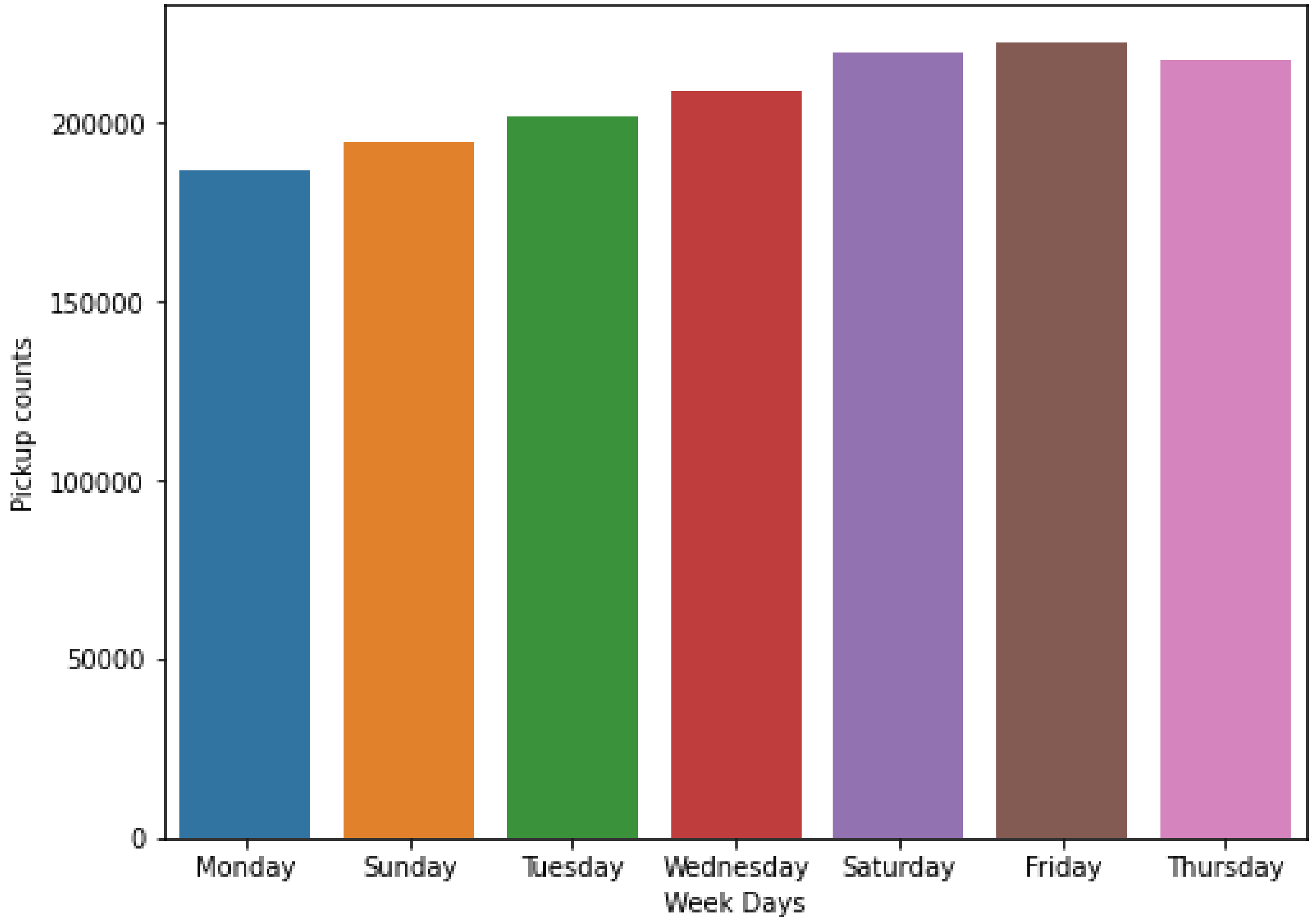


From the above two plots we can notice that most of the pickups and drops are between 6 PM to 7 PM. least number of pickups are between 4 AM to 5 AM.

Univariate Analysis

Trips Per Day

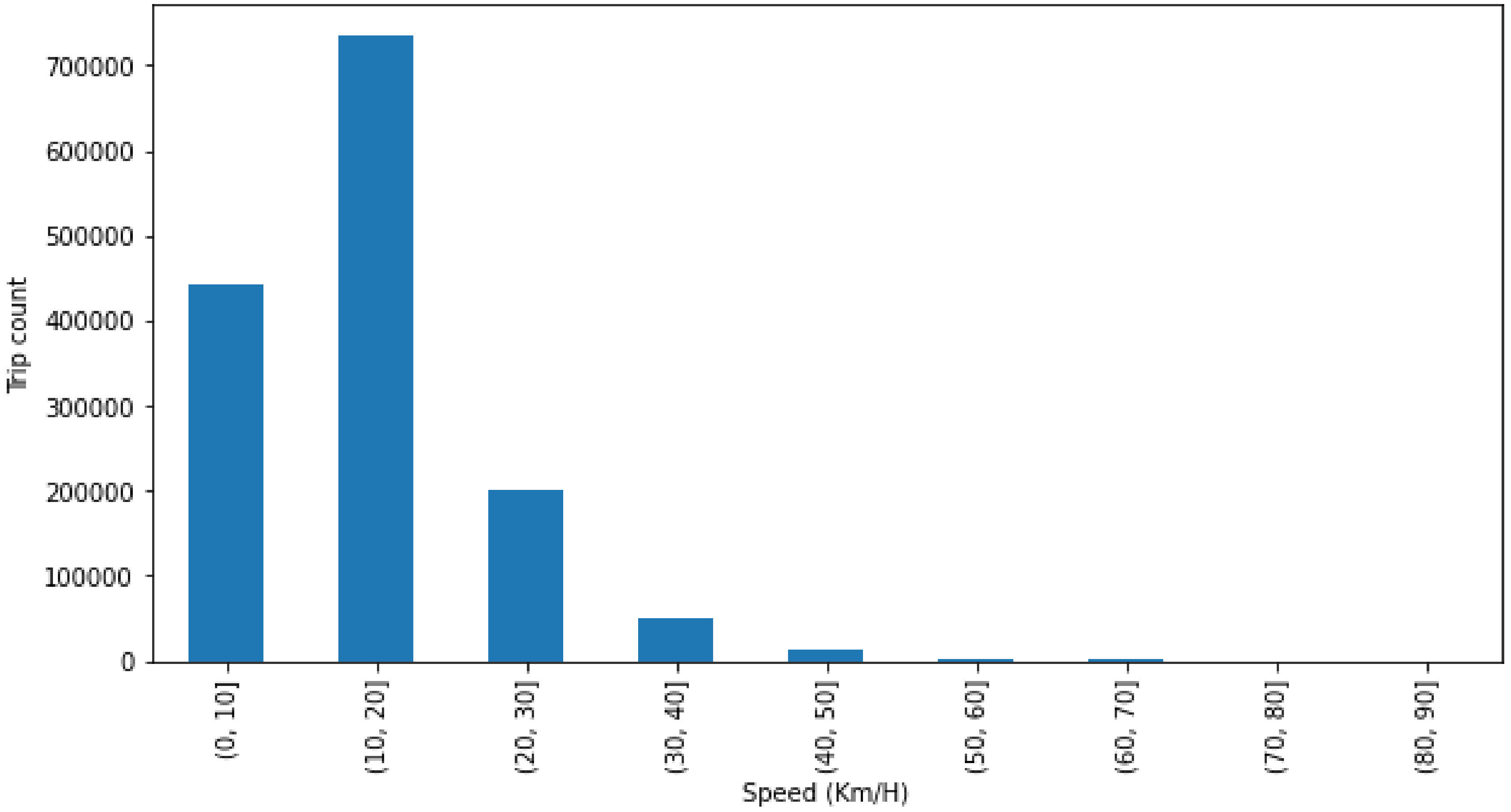
```
sns.countplot(data.pickup_weekday)
```



We can see the number of pickups starts increasing from monday to friday, then starts decreasing on weekends

Speed

```
data.speed.groupby(pd.cut(data.speed, np.arange(0,100,10))).count().plot(kind = 'bar')
```

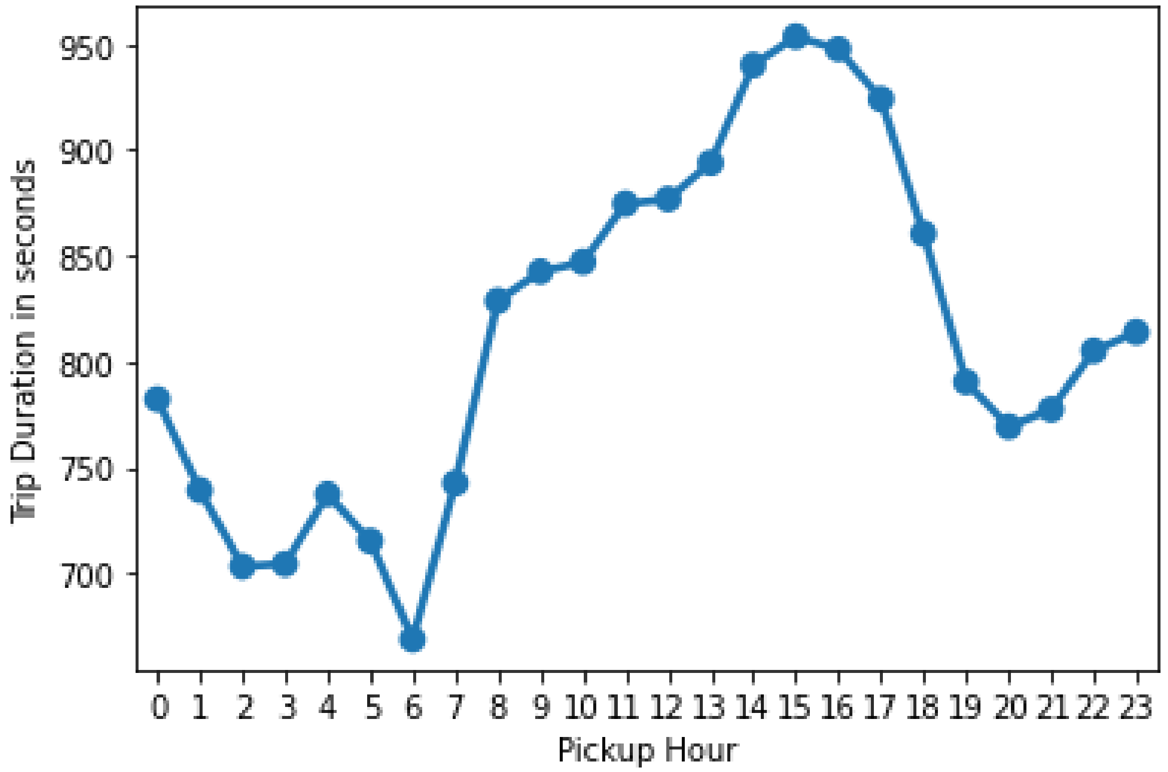


Most of the trips are done at the range 10-20 Km/Hr

Bivariate Analysis

Trip Duaration Per Hour

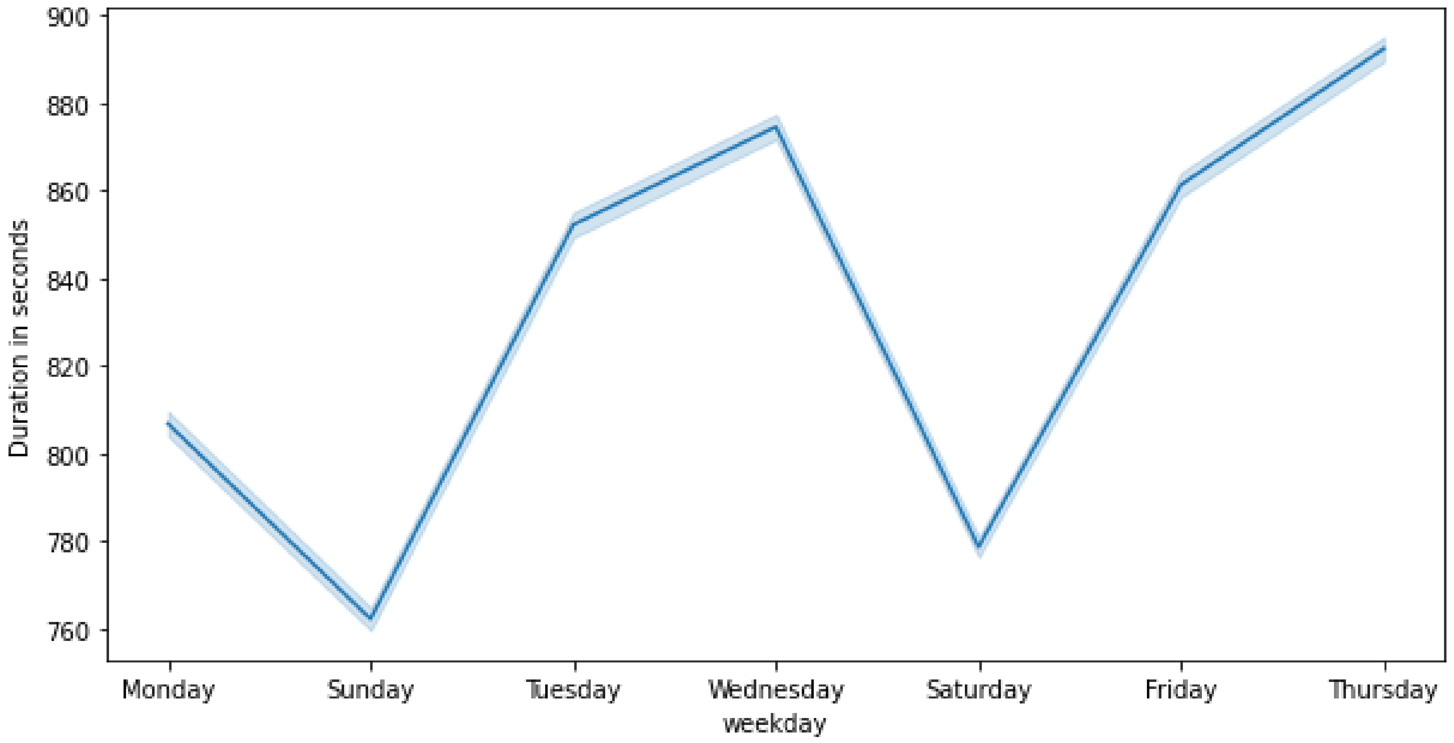
duration = data.groupby('pickup_hour').trip_duration.mean()
sns.pointplot(duration.index, duration.values)



- From above plot we can see lowest mean trip duration is at 6 AM.
- Highest mean trip duration is at 3 PM. Most probaly due to high traffic

Trip Duration per Weekday

sns.lineplot(x='pickup_weekday',y='trip_duration',data=data)

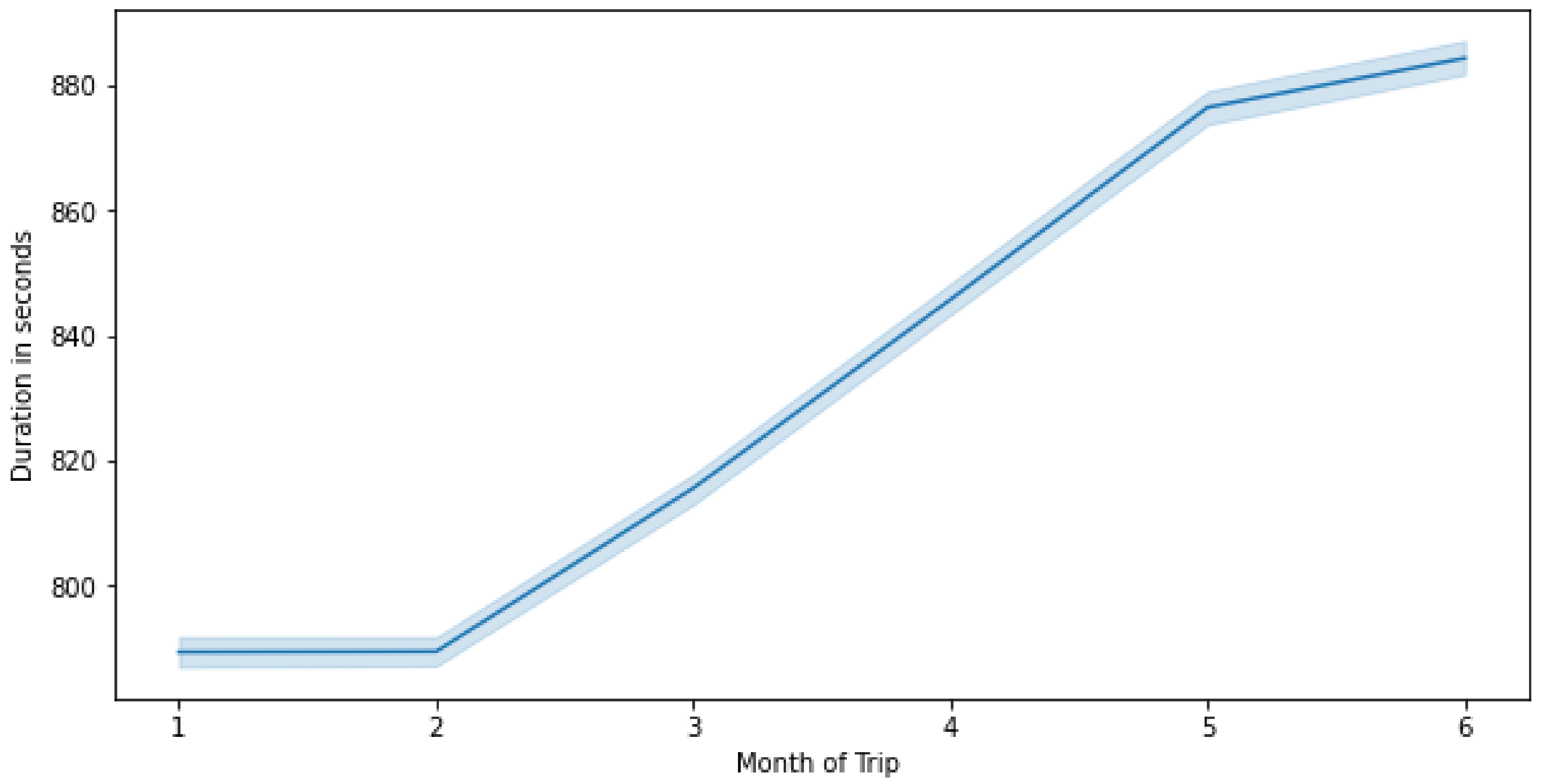


- Trip duration is highest on thursday
- Trip duration is lowest on weekends, because of holiday.

Bivariate Analysis

Trip Duaration Per Month

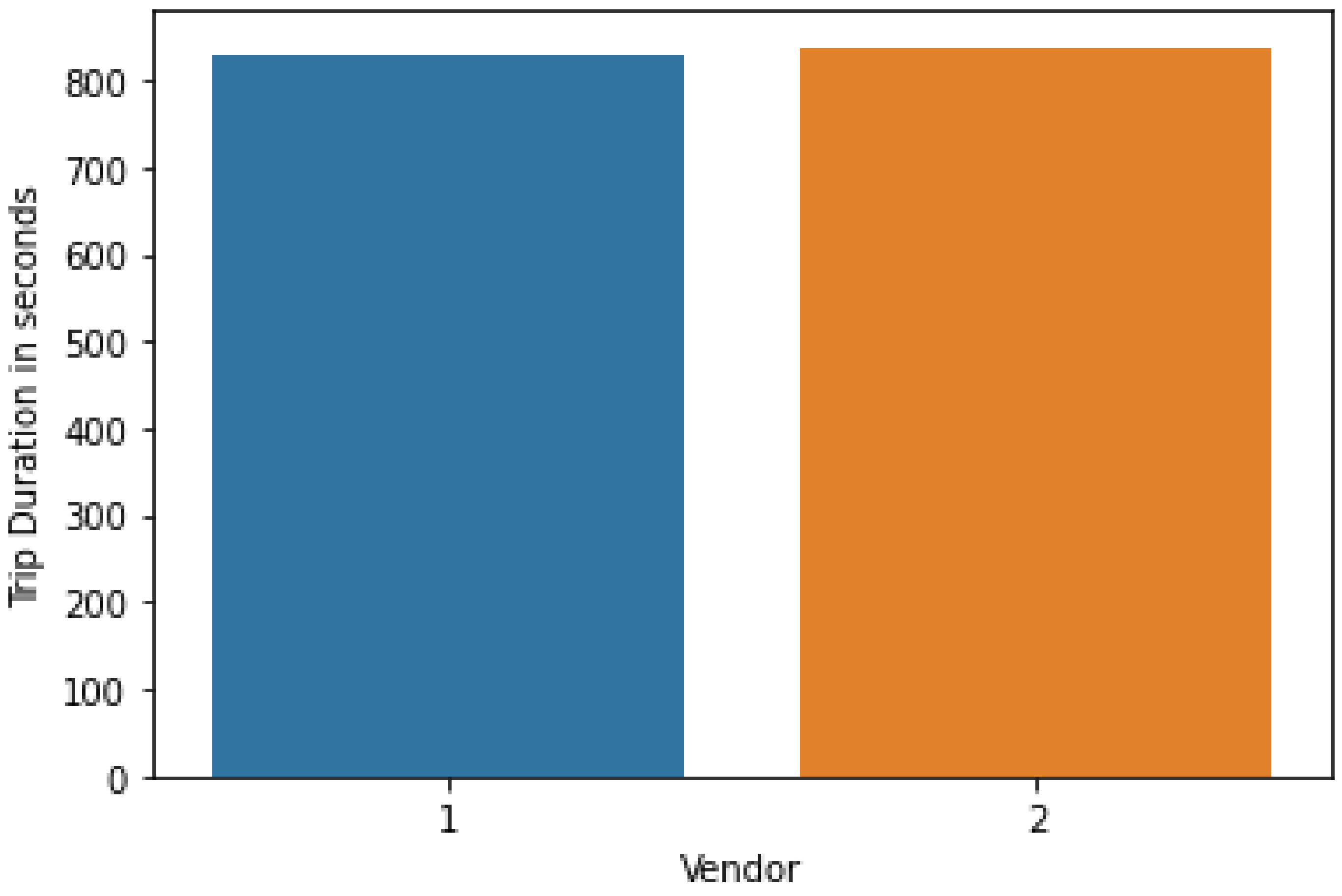
sns.lineplot(x='month',y='trip_duration', data=data)



- **There is an increasing trend of trip duration as month increases.**
- **This might be due to some climatic conditions like rain and snow.**
- **As in rainy season roads are more blocked and trip duration will increase**

Trip Duration per Vendor

sns.lineplot(x='pickup_weekday',y='trip_duration',data=data)

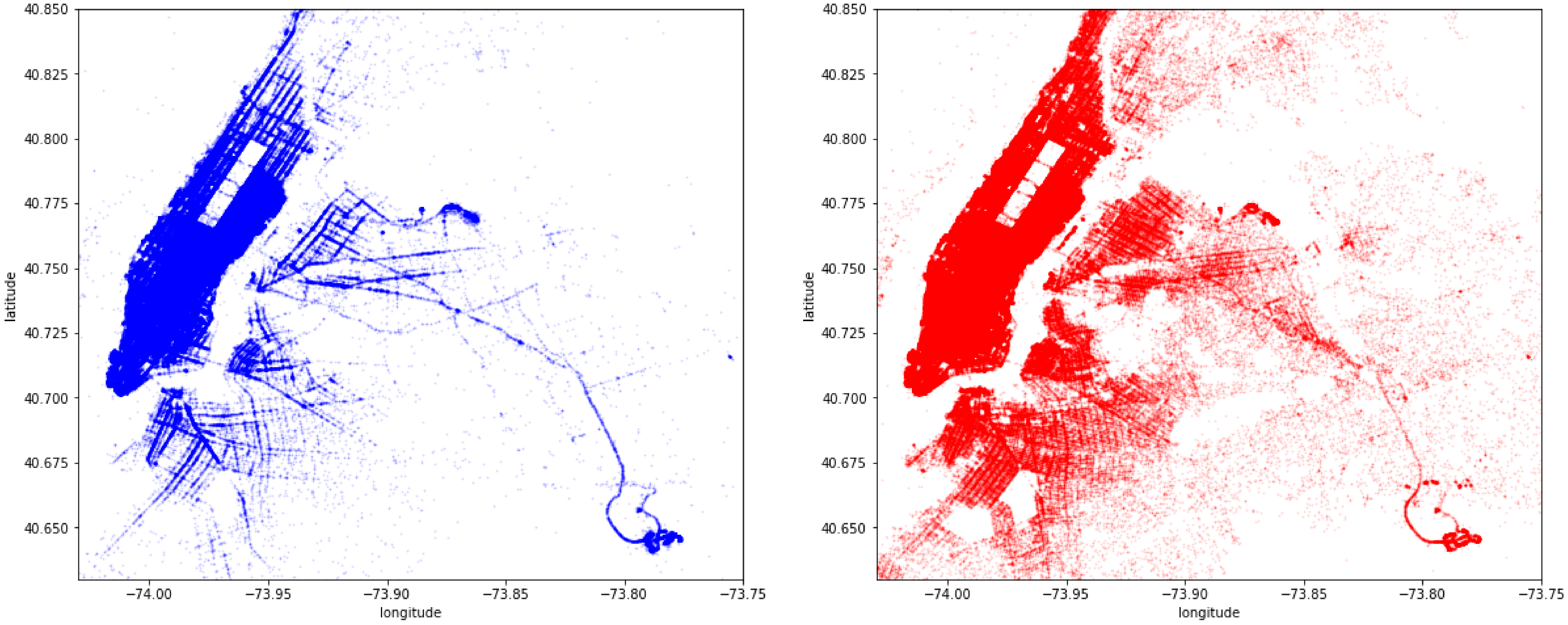


- **Trip duration for vendor 2 is higher than vendor 1**

Location Visualization On Map

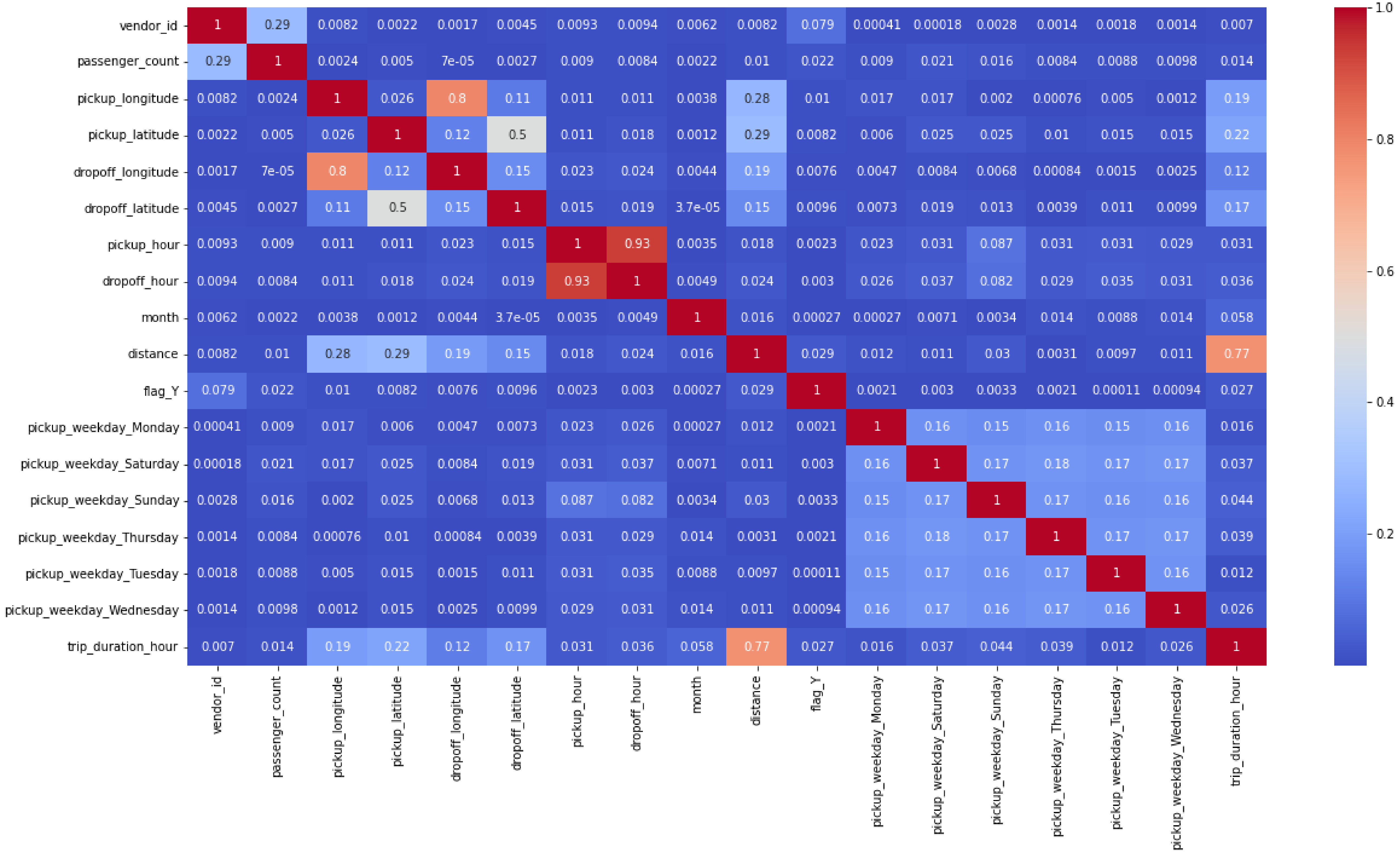
Plotting pickup and dropoff location according to their lattitude and longitude. Here we can see the map of the streets of NYC very clearly.

```
plt.scatter(data['pickup_longitude'].values, data['pickup_latitude'].values,
color='blue', s=1, alpha=0.1)
```



Correlation Heatmap

sns.heatmap(abs(corelation), annot=True, cmap='coolwarm')



	vendor_id	passenger_count	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	pickup_hour	dropoff_hour	month	distance	flag_Y	pickup_weekday_Monday	pickup_weekday_Saturday	pickup_weekday_Sunday	pickup_weekday_Thursday	pickup_weekday_Tuesday	pickup_weekday_Wednesday	trip_duration_hour
vendor_id	1	0.29	0.0082	0.0022	0.0017	0.0045	0.0093	0.0094	0.0062	0.0082	0.079	0.00041	0.00018	0.0028	0.0014	0.0018	0.0014	0.007
passenger_count	0.29	1	0.0024	0.005	7e-05	0.0027	0.009	0.0084	0.0022	0.01	0.022	0.009	0.021	0.016	0.0084	0.0088	0.0098	0.014
pickup_longitude	0.0082	0.0024	1	0.026	0.8	0.11	0.011	0.011	0.0038	0.28	0.01	0.017	0.017	0.002	0.00076	0.005	0.0012	0.19
pickup_latitude	0.0022	0.005	0.026	1	0.12	0.5	0.011	0.018	0.0012	0.29	0.0082	0.006	0.025	0.025	0.01	0.015	0.015	0.22
dropoff_longitude	0.0017	7e-05	0.8	0.12	1	0.15	0.023	0.024	0.0044	0.19	0.0076	0.0047	0.0084	0.0068	0.00084	0.0015	0.0025	0.12
dropoff_latitude	0.0045	0.0027	0.11	0.5	0.15	1	0.015	0.019	3.7e-05	0.15	0.0096	0.0073	0.019	0.013	0.0039	0.011	0.0099	0.17
pickup_hour	0.0093	0.009	0.011	0.011	0.023	0.015	1	0.93	0.0035	0.018	0.0023	0.023	0.031	0.087	0.031	0.031	0.029	0.031
dropoff_hour	0.0094	0.0084	0.011	0.018	0.024	0.019	0.93	1	0.0049	0.024	0.003	0.026	0.037	0.082	0.029	0.035	0.031	0.036
month	0.0062	0.0022	0.0038	0.0012	0.0044	3.7e-05	0.0035	0.0049	1	0.016	0.00027	0.00027	0.0071	0.0034	0.014	0.0088	0.014	0.058
distance	0.0082	0.01	0.28	0.29	0.19	0.15	0.018	0.024	0.016	1	0.029	0.012	0.011	0.03	0.0031	0.0097	0.011	0.77
flag_Y	0.079	0.022	0.01	0.0082	0.0076	0.0096	0.0023	0.003	0.00027	0.029	1	0.0021	0.003	0.0033	0.0021	0.00011	0.00094	0.027
pickup_weekday_Monday	0.00041	0.009	0.017	0.006	0.0047	0.0073	0.023	0.026	0.00027	0.012	0.0021	1	0.16	0.15	0.16	0.15	0.16	0.016
pickup_weekday_Saturday	0.00018	0.021	0.017	0.025	0.0084	0.019	0.031	0.037	0.0071	0.011	0.003	0.16	1	0.17	0.18	0.17	0.17	0.037
pickup_weekday_Sunday	0.0028	0.016	0.002	0.025	0.0068	0.013	0.087	0.082	0.0034	0.03	0.0033	0.15	0.17	1	0.17	0.16	0.16	0.044
pickup_weekday_Thursday	0.0014	0.0084	0.00076	0.01	0.00084	0.0039	0.031	0.029	0.014	0.0031	0.0021	0.16	0.18	0.17	1	0.17	0.17	0.039
pickup_weekday_Tuesday	0.0018	0.0088	0.005	0.015	0.0015	0.011	0.031	0.035	0.0088	0.0097	0.00011	0.15	0.17	0.16	0.17	1	0.16	0.012
pickup_weekday_Wednesday	0.0014	0.0098	0.0012	0.015	0.0025	0.0099	0.029	0.031	0.014	0.011	0.00094	0.16	0.17	0.16	0.17	0.16	1	0.026
trip_duration_hour	0.007	0.014	0.19	0.22	0.12	0.17	0.031	0.036	0.058	0.77	0.027	0.016	0.037	0.044	0.039	0.012	0.026	1

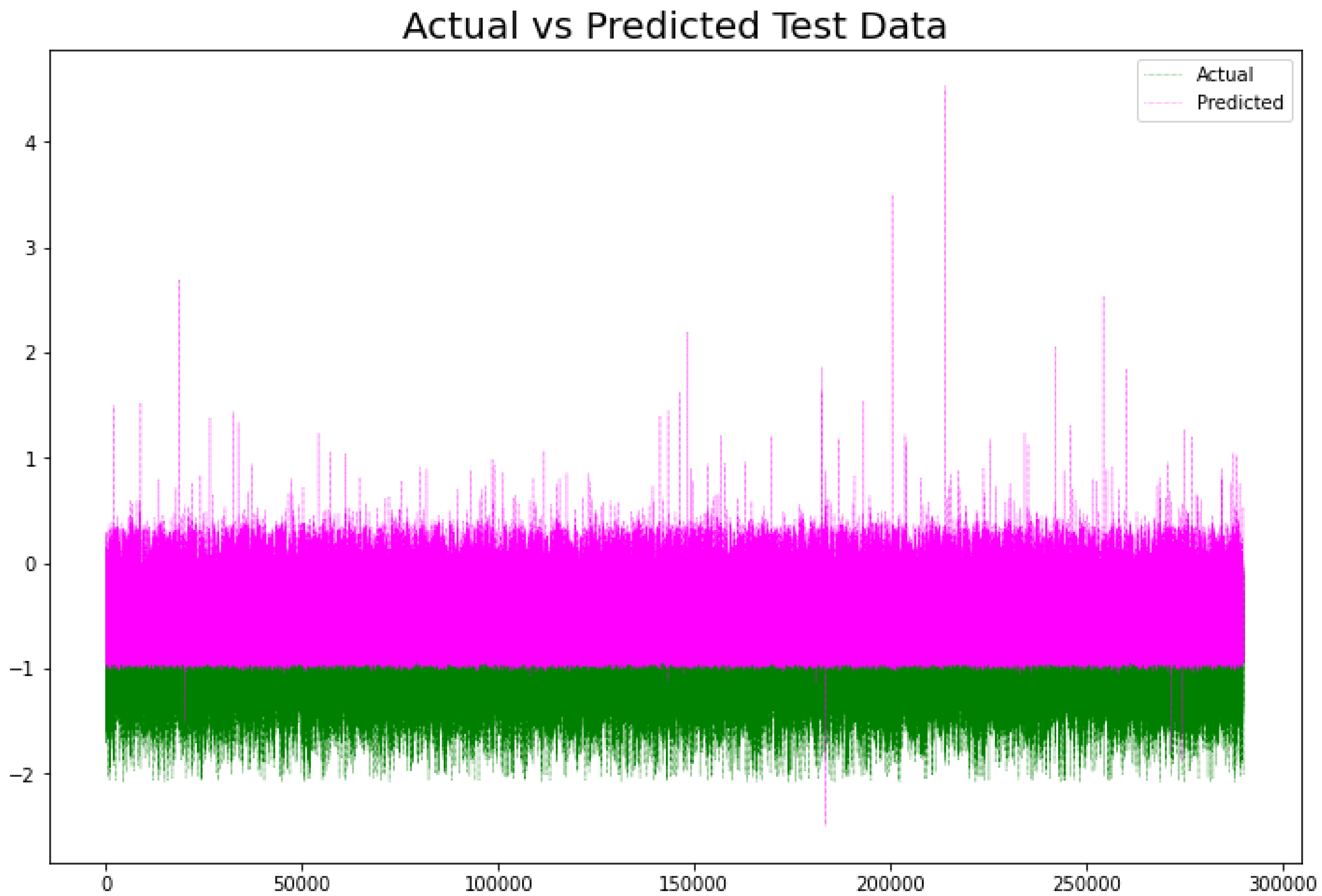
we can see that there isn't much relation between the variables except distance-trip duration and dropoff longitude and pickup longitude.

Linear Regression

Model Evaluation

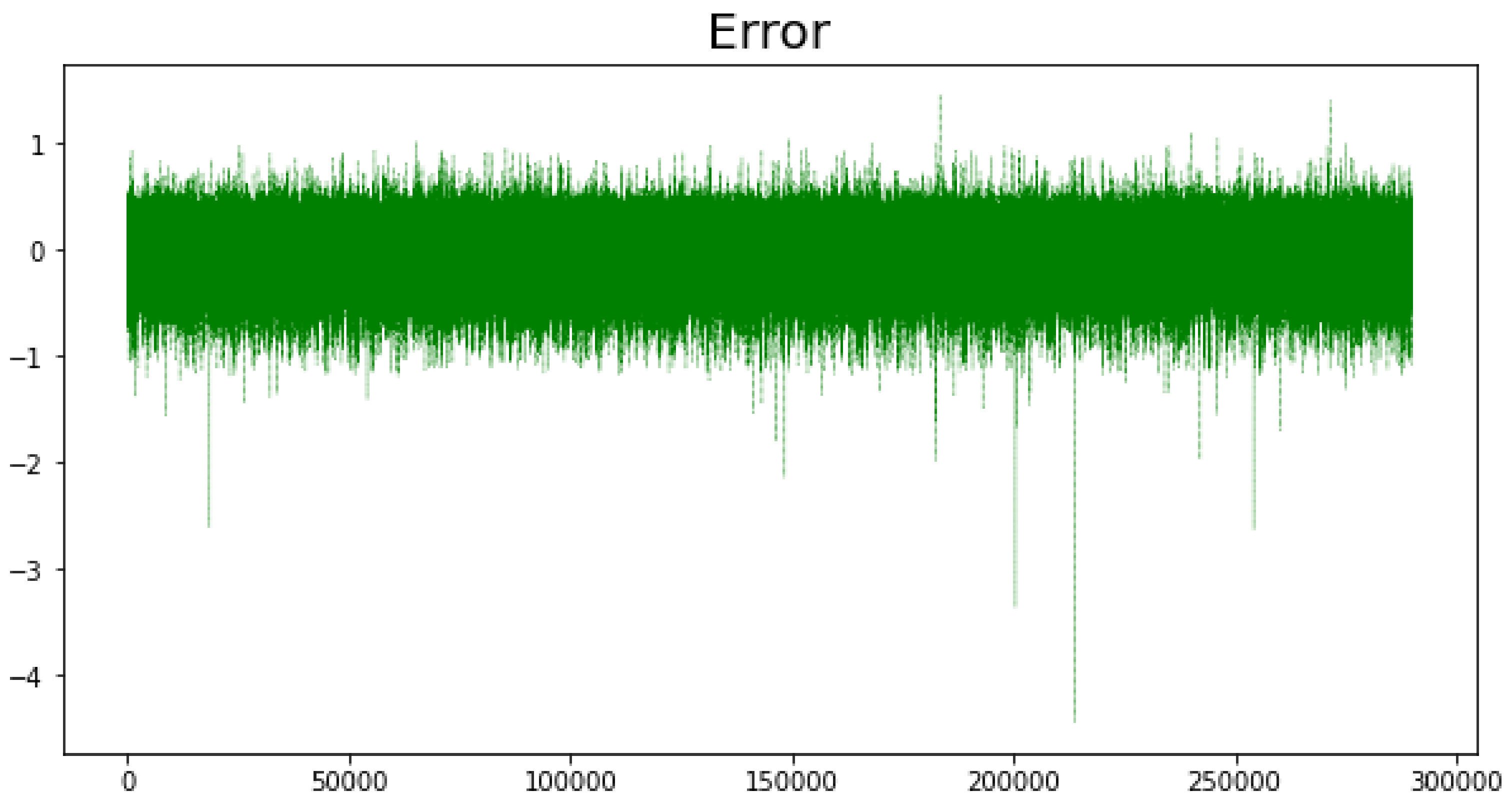
Visualization of Actual vs Predicted Data

```
plt.figure(figsize= (12,8))
c= [i for i in range(0, len(y_test))]
plt.plot(c, y_test, color='green', linewidth=0.5, linestyle=':')
plt.plot(c, y_pred_test, color='magenta', linewidth=0.5, linestyle=':')
plt.title('Actual vs Predicted Test Data', fontsize=20)
plt.legend(["Actual", "Predicted"])
```



Ploting the error

```
plt.figure(figsize= (10,5))
c= [i for i in range(0, len(y_test))]
plt.plot(c, y_test-y_pred_test, color='green', linewidth=0.5, linestyle=':')
plt.title('Error', fontsize=20)
plt.show()
```



Lasso Regression

Model Evaluation

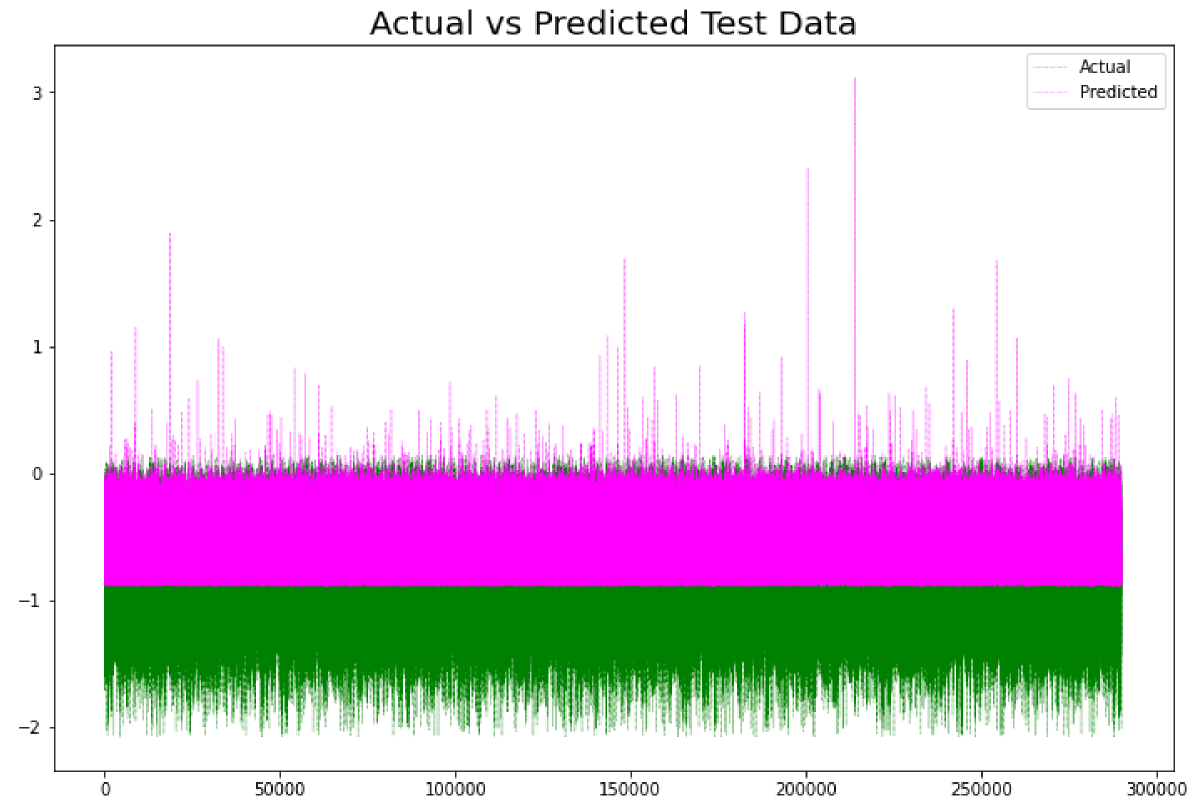
MSE RMSE R2 ADJUSTED R2 SCORE FOR TRAINING DATASET

Train MSE : 0.06132640952105937
Train RMSE : 0.24764169584514512
Train R2 : 0.40403416099387746
Train Adjusted R2 : 0.40402645457228004

MSE RMSE R2 ADJUSTED R2 SCORE FOR Test DATASET

Test MSE : 0.06128689967297325
Test RMSE : 0.24756191078793452
Test R2 : 0.40381719983745323
Test Adjusted R2 : 0.40378636173268256

Ploting the Actual vs Predicted data



XGBoost Model

Model Evaluation

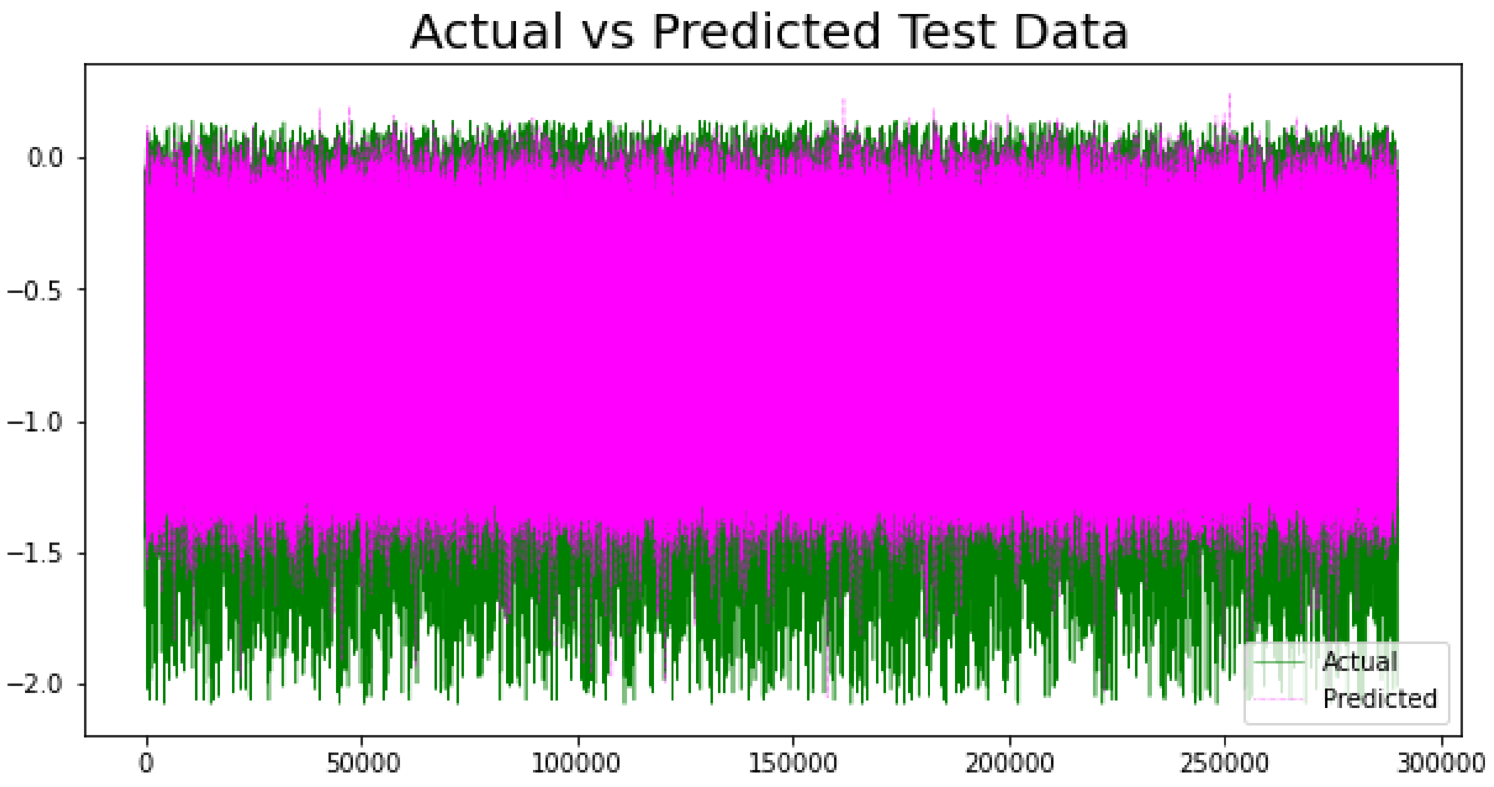
MSE RMSE R2 ADJUSTED R2 SCORE FOR TRAINING DATASET

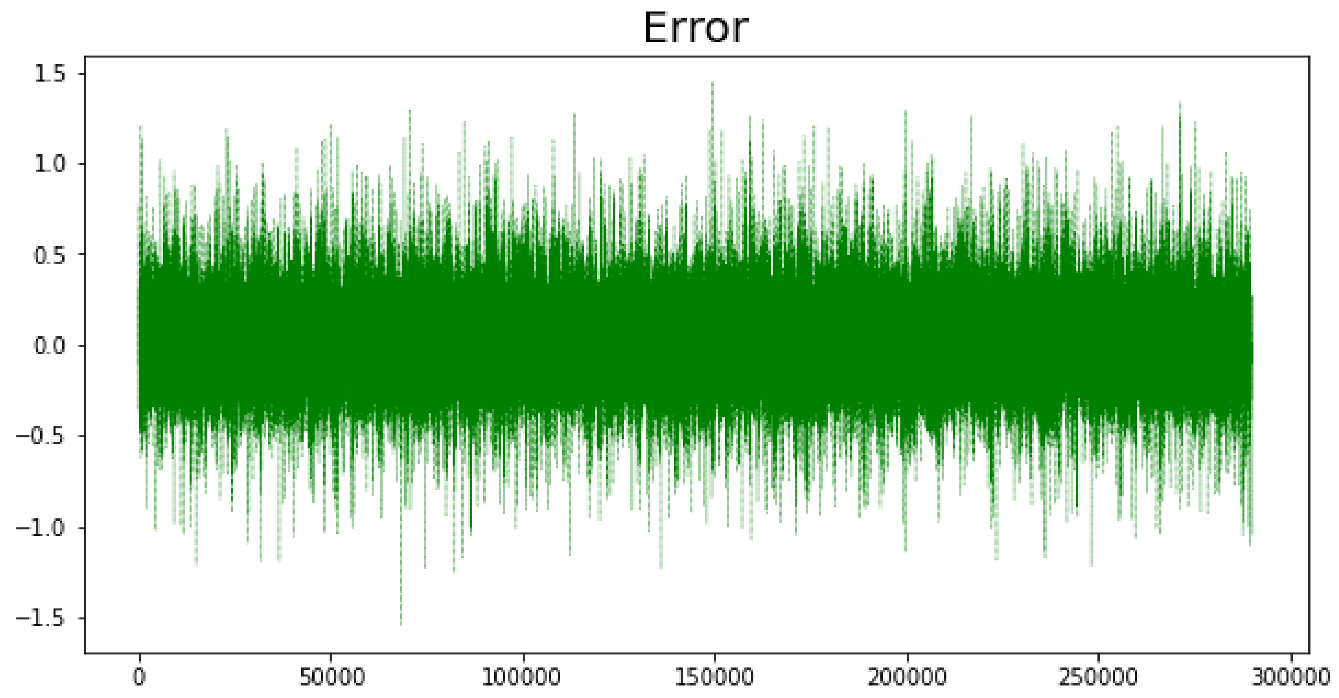
Train MSE : 0.017228304970182916
Train RMSE : 0.1312566378137994
Train R2 : 0.8325765146468818
Train Adjusted R2 : 0.8325743496973511

MSE RMSE R2 ADJUSTED R2 SCORE FOR Test DATASET

Test MSE : 0.01970132651011081
Test RMSE : 0.1403614138932449
Test R2 : 0.8083506904674747
Test Adjusted R2 : 0.8083407772303183

Ploting graph for Actual vs Predicted data





LightGBM Model

Model Evaluation

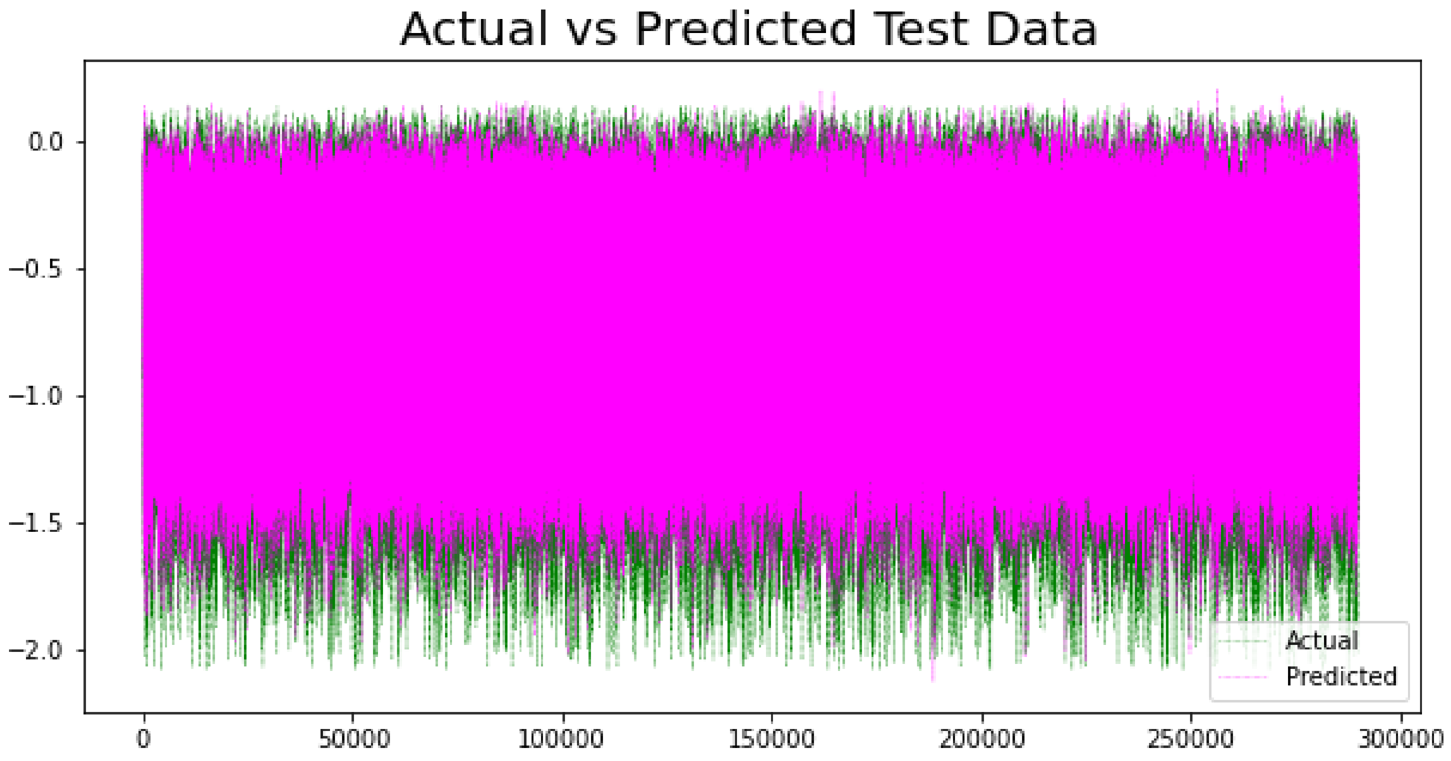
MSE RMSE R2 ADJUSTED R2 SCORE FOR TRAINING DATASET

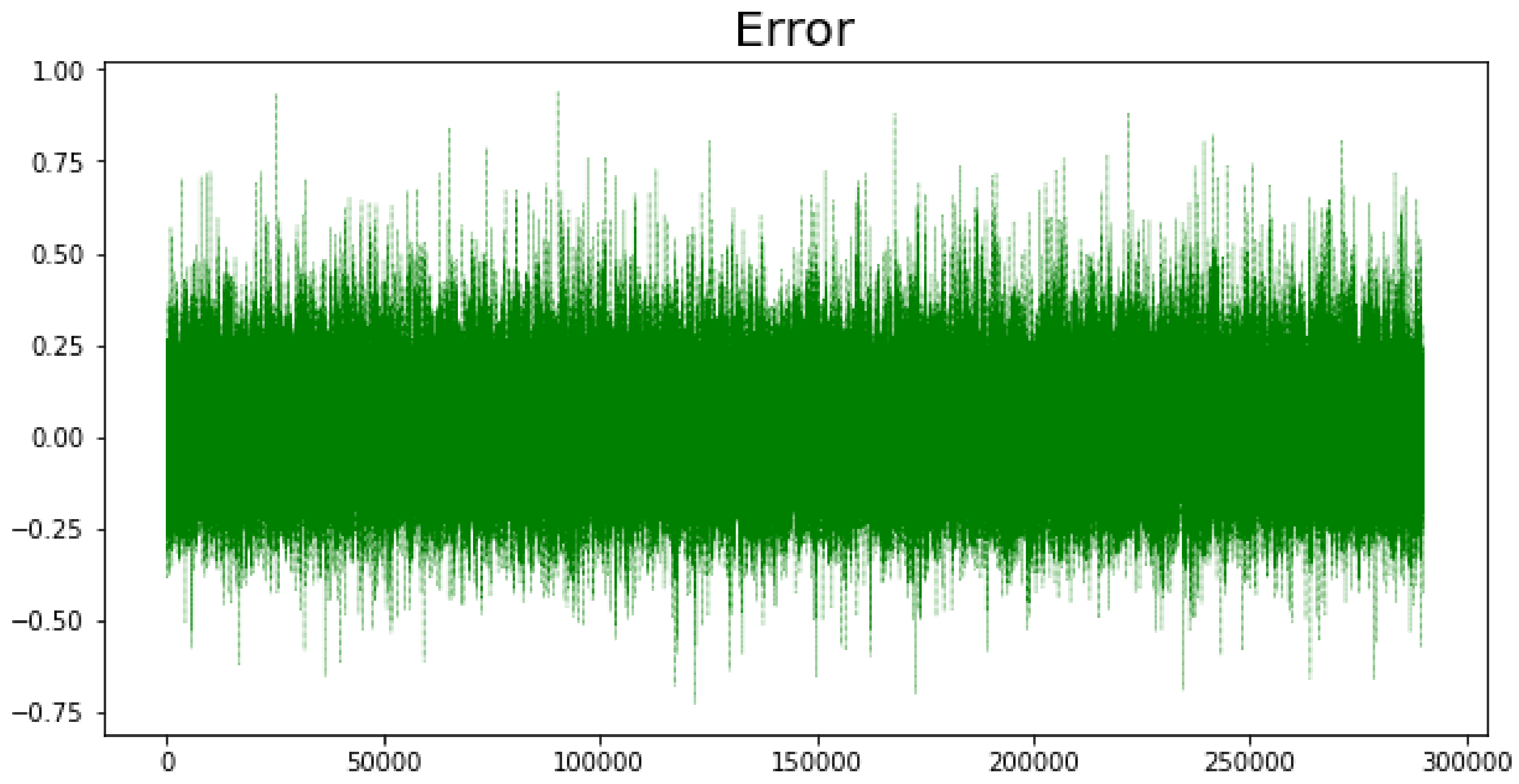
Train MSE : 0.010872571045036155
Train RMSE : 0.10427162147505022
Train R2 : 0.8943411007490413 Train
Adjusted R2 : 0.8943397344760607

MSE RMSE R2 ADJUSTED R2 SCORE FOR Test DATASET

Test MSE : 0.010934883632925212
Test RMSE : 0.10456999394149935
Test R2 : 0.8936283352802094
Test Adjusted R2 : 0.8936228331078213

Ploting graph for Actual vs Predicted data

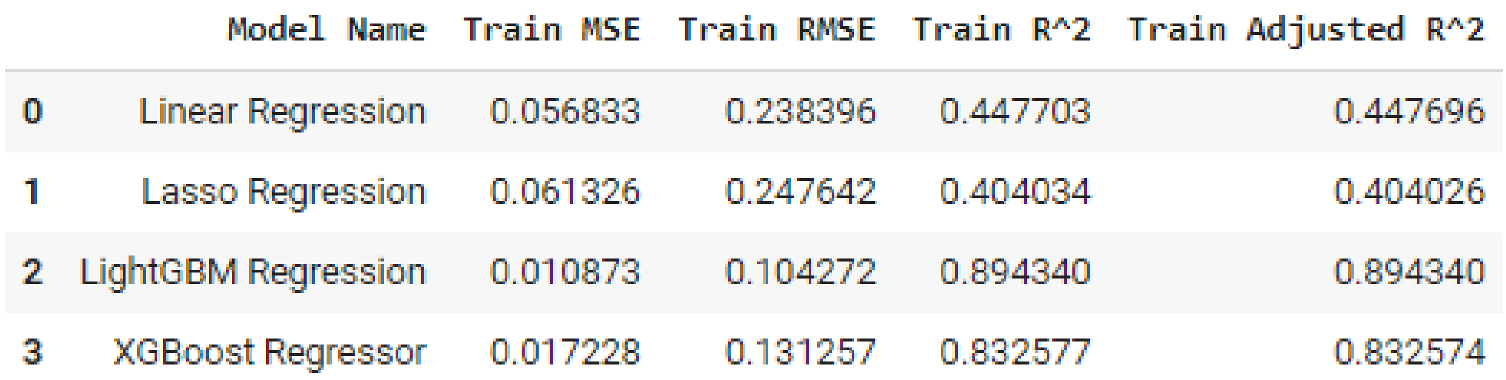




Model Summary

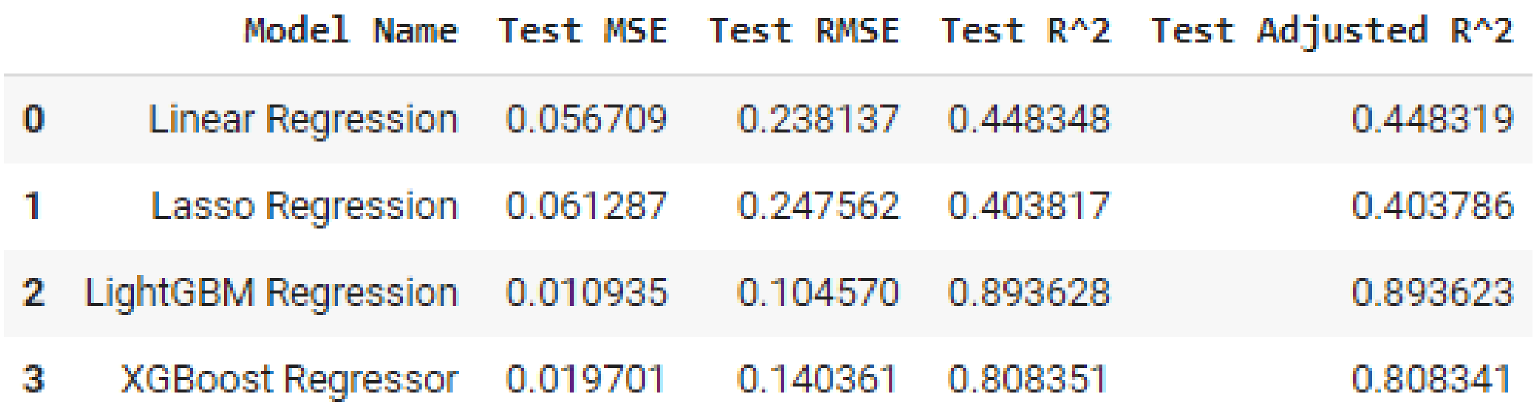
Here we compare different parameters of all models.

Train_data_df



	Model Name	Train MSE	Train RMSE	Train R^2	Train Adjusted R^2
0	Linear Regression	0.056833	0.238396	0.447703	0.447696
1	Lasso Regression	0.061326	0.247642	0.404034	0.404026
2	LightGBM Regression	0.010873	0.104272	0.894340	0.894340
3	XGBoost Regressor	0.017228	0.131257	0.832577	0.832574

Test_data_df



	Model Name	Test MSE	Test RMSE	Test R^2	Test Adjusted R^2
0	Linear Regression	0.056709	0.238137	0.448348	0.448319
1	Lasso Regression	0.061287	0.247562	0.403817	0.403786
2	LightGBM Regression	0.010935	0.104570	0.893628	0.893623
3	XGBoost Regressor	0.019701	0.140361	0.808351	0.808341

Conclusion

Comparing different parameters of all models.

- We compared MSE, RMSE and R2 for all four regression models, to find which is the best model to predict the NYC taxi trip duration.
- The Linear Regression and Lasso Regression didn't show any good prediction as compared to the other two.

- From he comparison table we can clearly see that XGBoost and LightGBM are the best models to predict trip duration of theNYC taxi. While LightGBM is fastest and more accurate than XGBoost. So, in between these two LightGBM is the best model.

- R Square: R2 score represents the coefficient of how well the values fit compared to the original values. The value from 0 to 1 interpreted as percentages. The higher the value is, the better the model is.

- RMSE (Root Mean Squared Error): RMSE is the standard deviation of the residuals. Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it displays how concentrated the data is around the line of best fit.

Thank you