Capstone Project - 2

NYC Taxi Trip Time Prediction

Individual Project by:

Anish Johnson

Contents:

1. Introduction.
2. Objective.
3. Dataset preview.
4. Exploratory Data Analysis.
5. Data Cleaning And Feature Engineering.
6. Data Preparation.
7. Model Building And Selection.
8. Conclusions

Introduction:

Taxi services play an important role in daily commute for people in NYC, according to wikipedia 1.6% of the overall population rely on taxis for their daily travelling.

With the increasing use of taxis companies try to provide their services as fast as possible. These services do come at a cost as they collect data and analyse it to find the factors that affect the trip durations, which then help in predicting the same.

Objective:

Our task is to build a model that predicts the total ride duration of taxi trips in New York City.

For which we have been provided a primary dataset which is released by the NYC Taxi and Limousine Commission, which includes pickup time, geo-coordinates, number of passengers, and several other variables.

Dataset Preview:

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.

Let's have a look at these features….

Dependent Feature:

- **trip_duration**: duration of the trip in seconds.

Independent Features:

- **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

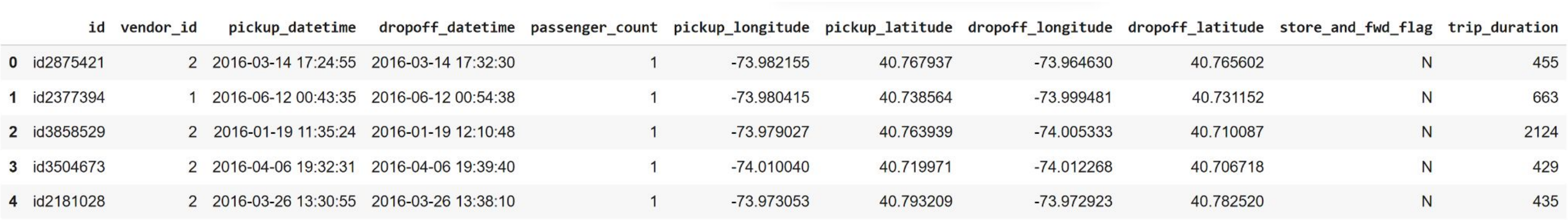
Exploratory Data Analysis:

What is Exploratory Data Analysis?

Simply defined, EDA is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations.

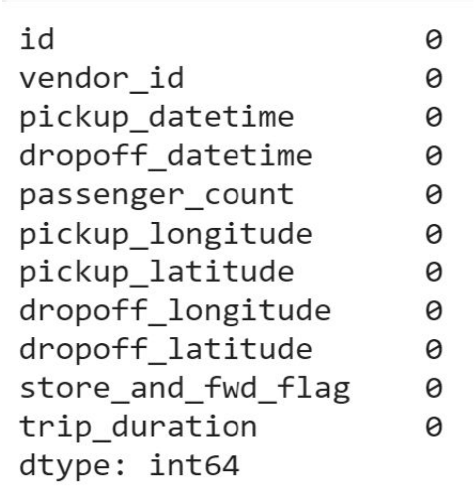
Why is EDA important?

As said by David McCandless, “Visualizing information can give us a very quick solution to problems. We can get clarity or the answer to a simple problem very quickly.” So let's do some visualization….



	id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration
0	id2875421	2	2016-03-14 17:24:55	2016-03-14 17:32:30	1	-73.982155	40.767937	-73.964630	40.765602	N	455
1	id2377394	1	2016-06-12 00:43:35	2016-06-12 00:54:38	1	-73.980415	40.738564	-73.999481	40.731152	N	663
2	id3858529	2	2016-01-19 11:35:24	2016-01-19 12:10:48	1	-73.979027	40.763939	-74.005333	40.710087	N	2124
3	id3504673	2	2016-04-06 19:32:31	2016-04-06 19:39:40	1	-74.010040	40.719971	-74.012268	40.706718	N	429
4	id2181028	2	2016-03-26 13:30:55	2016-03-26 13:38:10	1	-73.973053	40.793209	-73.972923	40.782520	N	435

- The first 5 values of our data gives us a basic idea of what we need to work on.

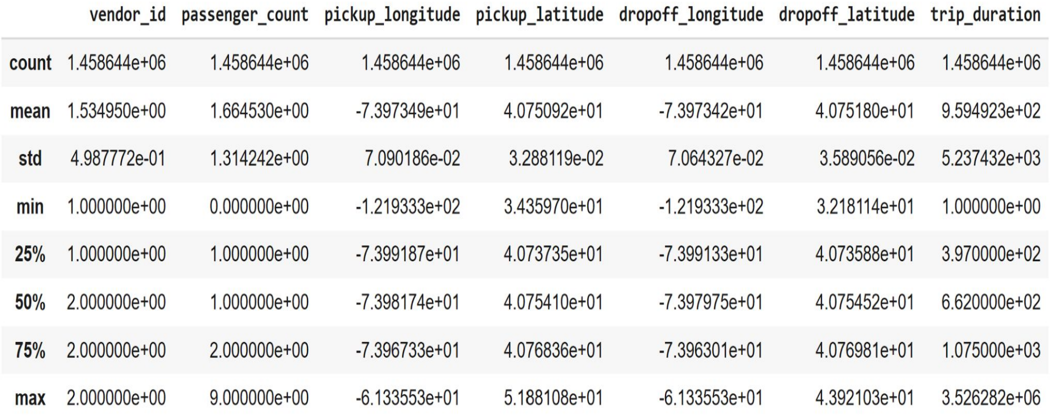


```
id                    0
vendor_id             0
pickup_datetime       0
dropoff_datetime      0
passenger_count       0
pickup_longitude      0
pickup_latitude       0
dropoff_longitude     0
dropoff_latitude      0
store_and_fwd_flag    0
trip_duration         0
dtype: int64
```

- The check for null values revealed that our data doesn't contain any null values.
- The dataset contains 1458644 rows and 11 columns and none of these values are duplicates.

Points to noted from statistics of data:

- pickup_datetime and dropoff_time need to be changed to datetime format - currently they are in string (object) format.
- extract data from datetime values.

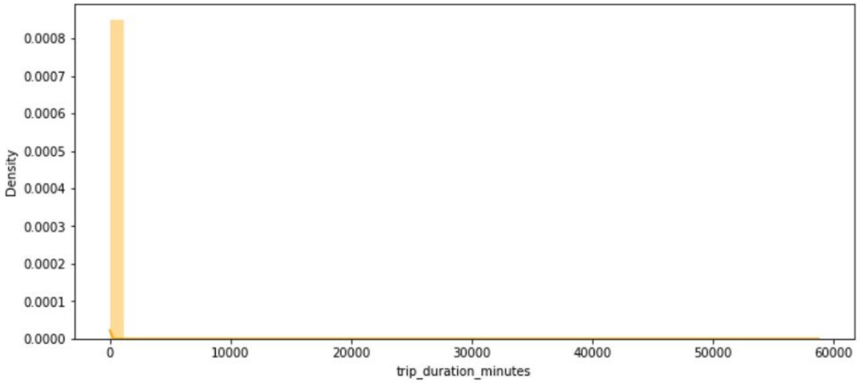


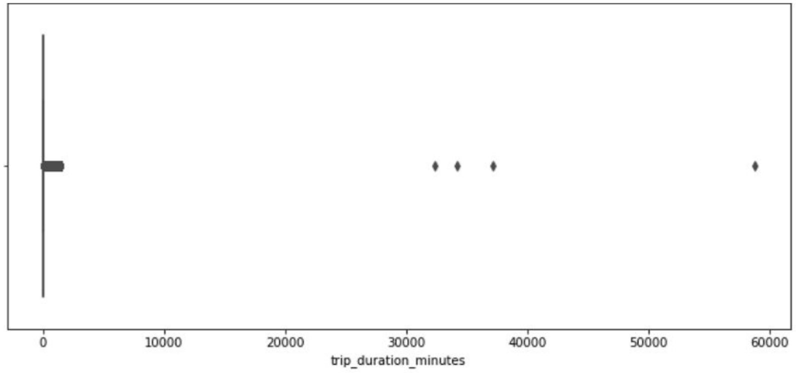
	vendor_id	passenger_count	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	trip_duration
count	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06
mean	1.534950e+00	1.664530e+00	-7.397349e+01	4.075092e+01	-7.397342e+01	4.075180e+01	9.594923e+02
std	4.987772e-01	1.314242e+00	7.090186e-02	3.288119e-02	7.064327e-02	3.589056e-02	5.237432e+03
min	1.000000e+00	0.000000e+00	-1.219333e+02	3.435970e+01	-1.219333e+02	3.218114e+01	1.000000e+00
25%	1.000000e+00	1.000000e+00	-7.399187e+01	4.073735e+01	-7.399133e+01	4.073588e+01	3.970000e+02
50%	2.000000e+00	1.000000e+00	-7.398174e+01	4.075410e+01	-7.397975e+01	4.075452e+01	6.620000e+02
75%	2.000000e+00	2.000000e+00	-7.396733e+01	4.076836e+01	-7.396301e+01	4.076981e+01	1.075000e+03
max	2.000000e+00	9.000000e+00	-6.133553e+01	5.188108e+01	-6.133553e+01	4.392103e+01	3.526282e+06

- trip_duration is given in seconds lets convert it into minutes.
- store_and_fwd_flag is a categorical variable that needs to be converted.
- vendor_id consists of two values 1 and 2.
- passenger_count ranges from 0-9, the difference between the 75th percentile and the max value shows the presence of outliers.
- trip_duration_minutes also contains outliers.

Dependent Variable:

- trip_duration_minutes is positively skewed.
- extreme values are present after 3000 minutes, which exceed more than 20 days, it was either due to some technical error or a vacation was planned.
- since there are only few values greater than 3000 minutes lets remove them.

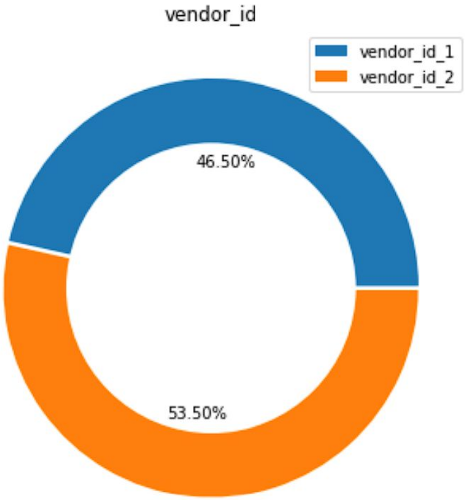


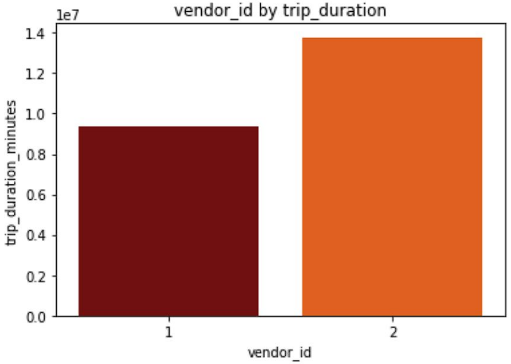


Independent Variables:

Vendor_id:

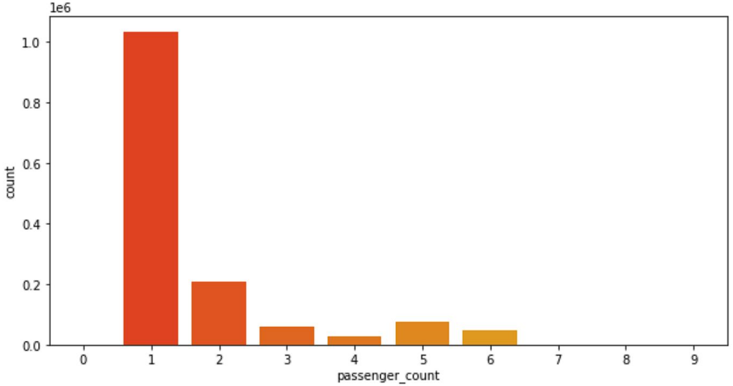
- 53.50% of the trips are completed by the vendor 2 and 46.50% of the trips are completed by vendor 1.
- also maximum duration is covered by vendor 2





Passenger_count:

- some values are zero which mean either the trip was cancelled or there was an error in the data entry.
- 7, 8, 9 are extreme cases considering the capacity of a car, so we will get rid of them.
- most of the trips (75%) contain at most 1 or 2 passengers.



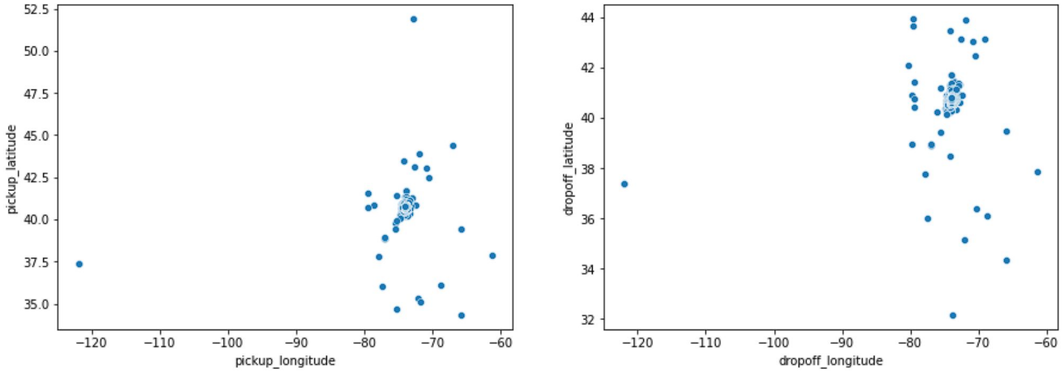
Pickup/Dropoff latitude and longitude:

- Most of the pickup and dropoff locations are situated in same area, but there are few values which act as outliers.

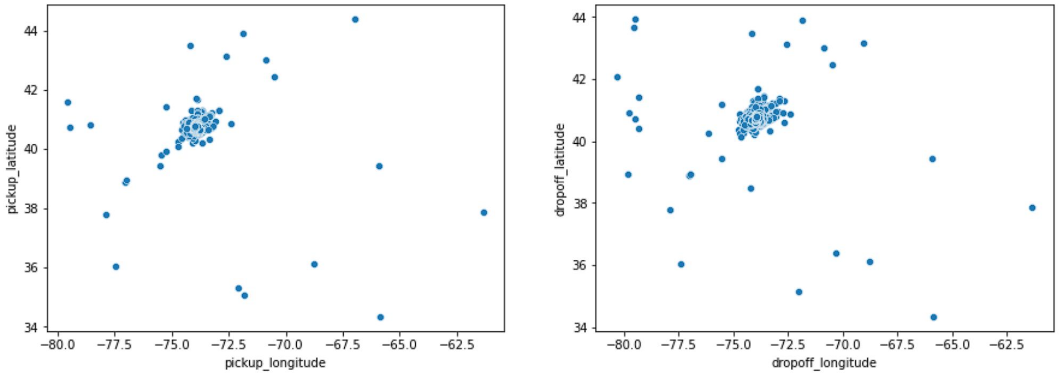
- On further analysis we found that id's [**id3777240, id2854272, id2306955, id0978162**] are the outliers which we will remove.

- After removing the outliers the plot looks much better.

Before removing outliers.



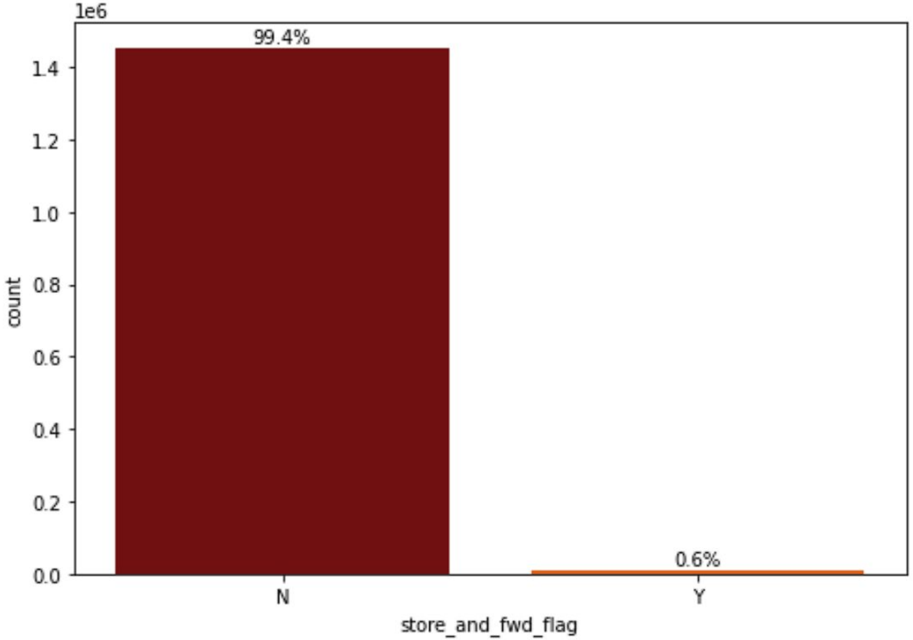
After removing outliers.



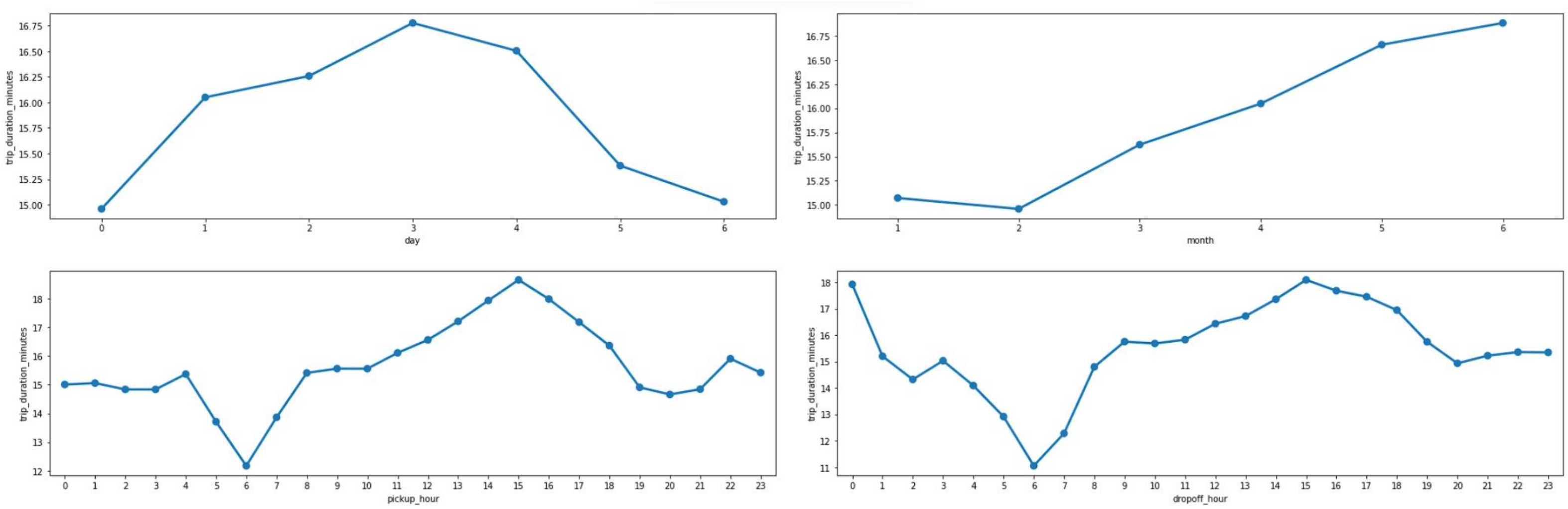
store_and_fwd_flag:

- 99.4% of the data values are N and only 0.6% 0f values are Y, which means most of the data was uploaded directly without storing it and forwarding.

- this ia a categorical feature which we will be converting into numeric by getting dummies.



day/month and pickup_hour/dropoff_hour:



- trip duration **decreases** as the **weekend** approaches, it makes sense as most of the people either stay at home or go for vacations.
- trip duration **increases** after **February**, this might be due to the people returning after holidays.
- pickup and dropoff hours are almost the same, most of the commute happen from **6am** to **7pm** after which it **gradually decreases**.

Data Cleaning And Feature Engineering:

In this step we will get rid of the outliers in our data and do some feature engineering to get better results.

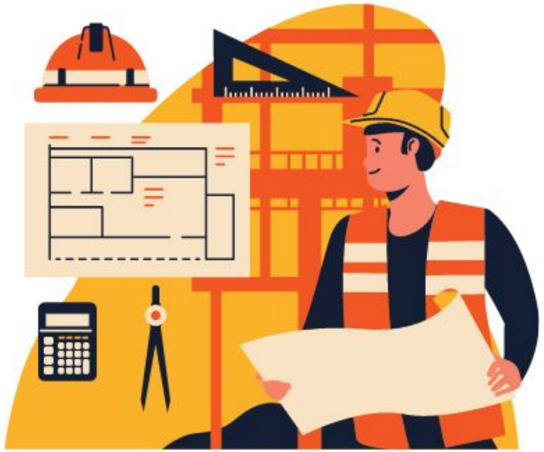
In order to remove outliers we will be using the Interquartile range for those features.

IQR = Q3 - Q1

lower_range = Q1 - (1.5 * IQR)
upper_range = Q3 - (1.5 * IQR)

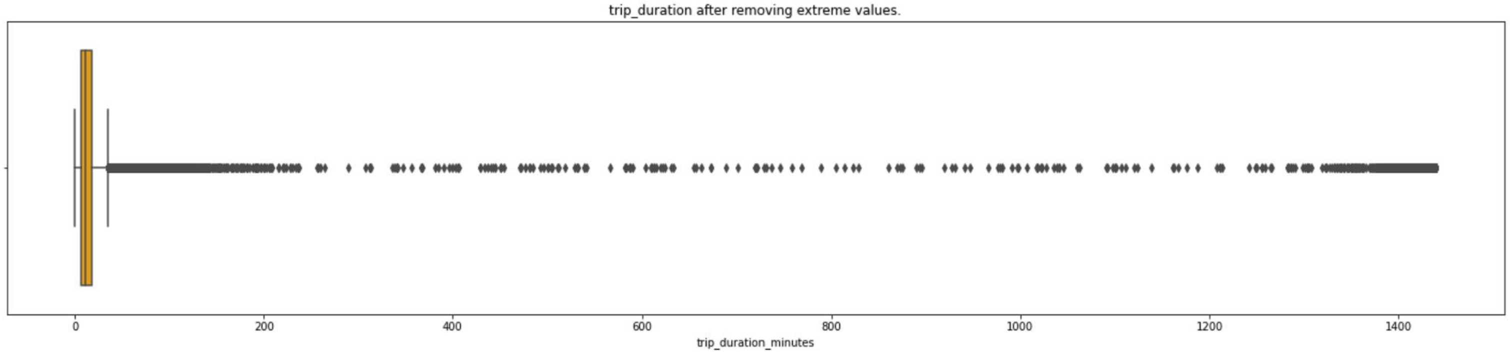
The IQR method involves calculating the lower and upper bounds for the data and removing the values that do not fall in this range, which indirectly removes the outliers.

And to do feature engineering we will utilize the given data to create new features that would be helpful in prediction.



Dependent Variable: trip_duration_minutes.

We start with removing the **outliers in the dependent variable**, to do so we will utilize the IQR method (Interquartile range) which removes the values which do not fall in the calculated limits.



lower_range for trip_duration_minutes = 10.333333333333336

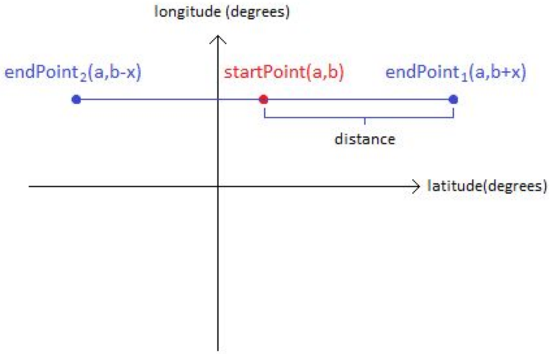
upper_range for trip_duration_minutes = 34.866666666666674

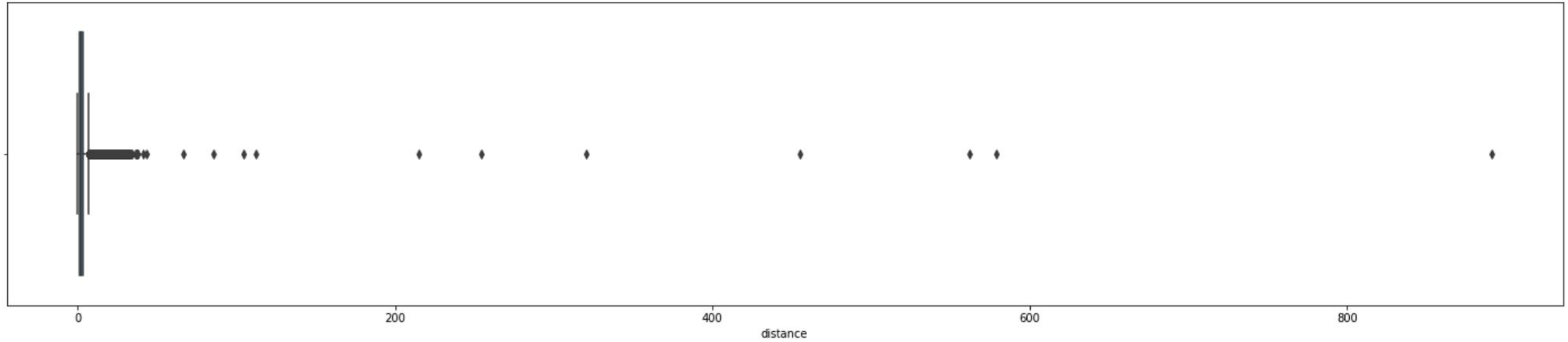
All the values that do not fall in these limits will be removed.

Now lets create a new feature for dataset. [distance]

We will use the longitude and latitude data to calculate the **distance** travelled between the pickup and dropoff location. To make it easier we used the haversine function to calculate the distance between the geographical values.

$$d = 2r \arcsin \sqrt{\sin^2 \frac{1}{2}(\phi_2 - \phi_1) + \cos \phi_1 \cos \phi_2 \sin^2 \frac{1}{2}(\lambda_2 - \lambda_1)}$$





There are many outliers in the variable that will be removed, we will use the IQR method to remove these outliers.

Removing outliers from distance.

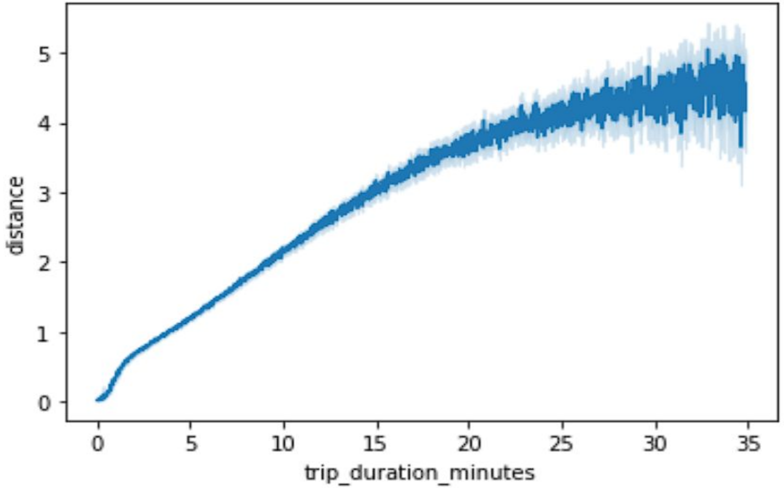
lower_range for trip_duration_minutes = -2.233443060186282

upper_range for trip_duration_minutes = 6.917934441398998

We will remove the values that do fall in these range.

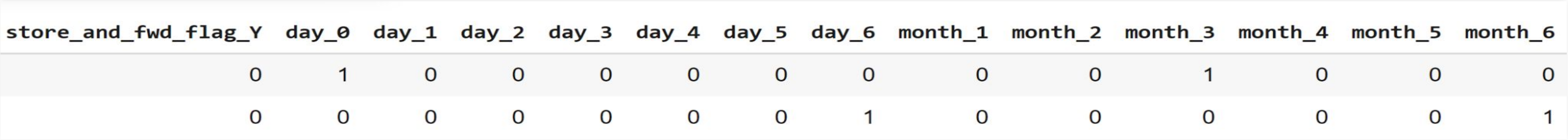
Now let's check relation between distance and trip_duration_minutes.

As we can see distance has a linear relationship with the dependent variable which is good for our model as we will be building regression models.



Get dummies for the categorical variables:

We got dummy variables for the categorical features: [**store_and_fwd_flag, day, month.**]



store_and_fwd_flag_Y	day_0	day_1	day_2	day_3	day_4	day_5	day_6	month_1	month_2	month_3	month_4	month_5	month_6
0	1	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	1

Transform pickup/ dropoff hours.

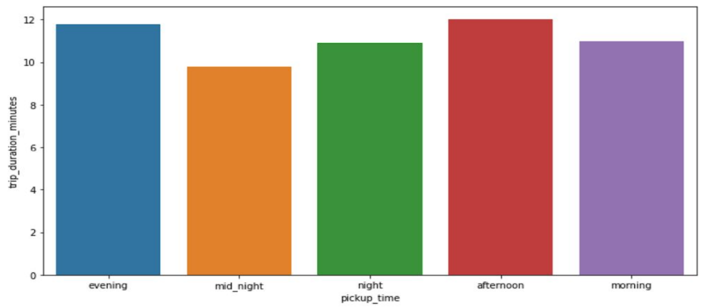
Since there are 24 different values in these two columns it would be better to categorize and get dummies for these variables. To do so we will split the hours into five categories:

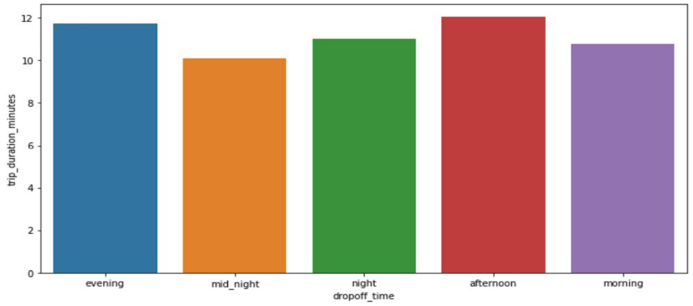
If between 0 and 5 = mid_night, If between 5 and 12 = morning

If between 12 and 16 = afternoon, If between 16 and 19 = evening

If between 19 and 24 = night

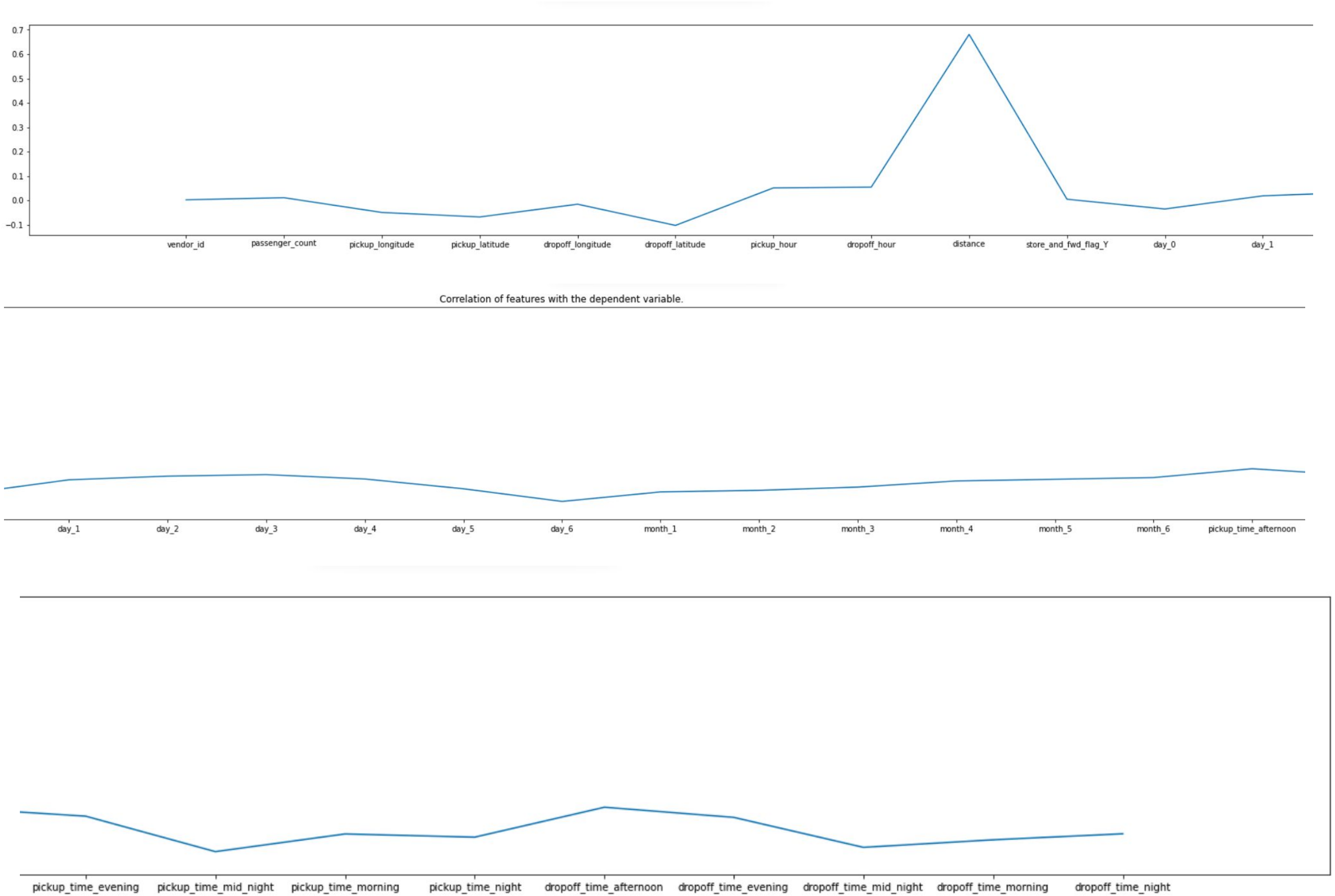
*most of the pickups and dropoffs happen in the **evenings** and **afternoons**, which makes sense as most people commute during these timings.*





Correlation with the dependent variable:

This plot makes clear which features are highly correlated with the dependent variable, we can observe that distance has highest correlation among all the features. **pickup_hour**, **dropoff_hour, distance**, **store_and_fwd_flag**, **day_1**, **day_2**, **day_3**, **day_4**, **day_5**, **pickup_time_afternoon**, **dropoff_time_afternoon** are more correlated with dependent feature than other features.

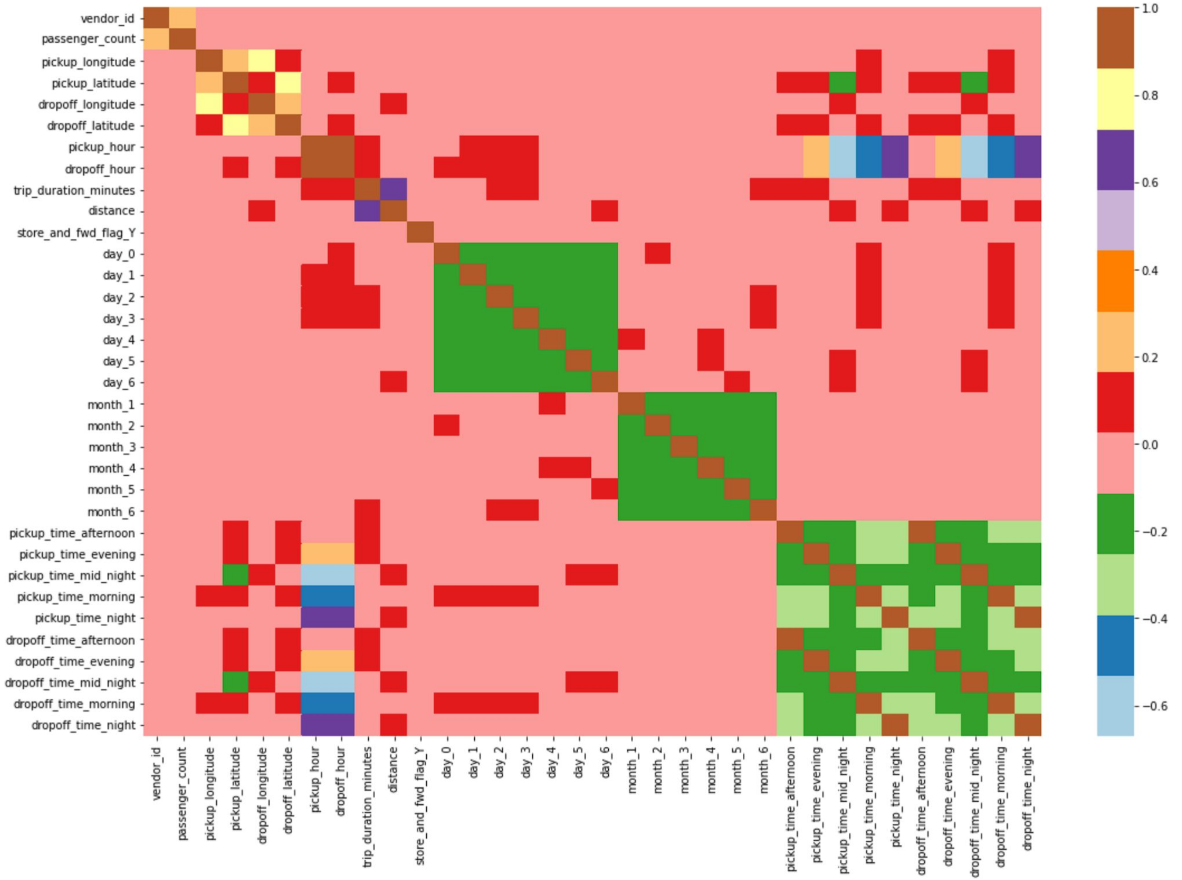


Correlation of features with the dependent variable.

Multicollinearity in features:

As we can see many of the features are correlated to each other, and among these the highly correlated groups are:
The **pickup/ dropoff longitude/ latitude data**, **pickup/ dropoff hours**.

Most of the **days**, **months** and **time categories** are negatively correlated to each other, the negative correlation between these features make sense as when one increases the other will decrease.



Data preparation:

We need to prepare the data before we put them through regression models.

We start by **dropping Id** from the data as it is no use to us.

Next separate the dependent and the independent variables, where y is the dependent variable trip_duration_minutes and X contains rest of the features in our dataset.

Now do the train test split to separate the training and the testing data that we will use to build and validate our regression models.

(80% training and 20% test data.)

Finally we will transform our data using Standard Scaler, this is done to standardize the data before feeding them to the models.

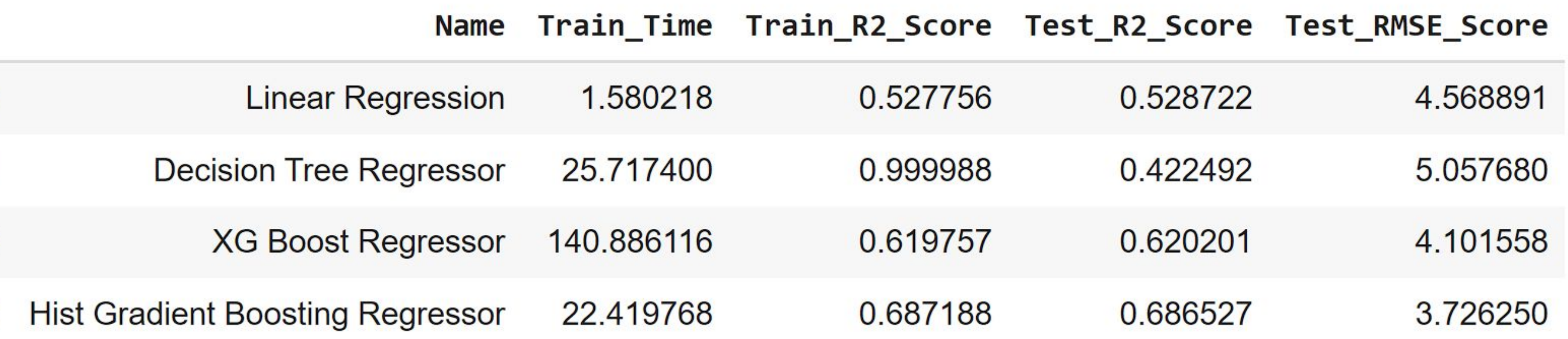
Model Building And Selection:

Now that we have prepared our data its time to build regression models using this data.

The models we will be using are:

- **Linear Regression**
- **Decision Tree Regressor**
- **XG Boost Regressor**
- **Hist Gradient Boosting Regressor**

We will compare these models and select the best performing model for the prediction.



Name	Train_Time	Train_R2_Score	Test_R2_Score	Test_RMSE_Score
Linear Regression	1.580218	0.527756	0.528722	4.568891
Decision Tree Regressor	25.717400	0.999988	0.422492	5.057680
XG Boost Regressor	140.886116	0.619757	0.620201	4.101558
Hist Gradient Boosting Regressor	22.419768	0.687188	0.686527	3.726250

As we can see from the above table Hist Gradient Boosting Regressor performs the best.

Hence we will proceed with it and try some hyperparameter tuning and cross validations to improve the score and reduce the error.

Hyperparameter Tuning:

To perform the hyperparameter tuning and cross validations we will use Grid Search CV.

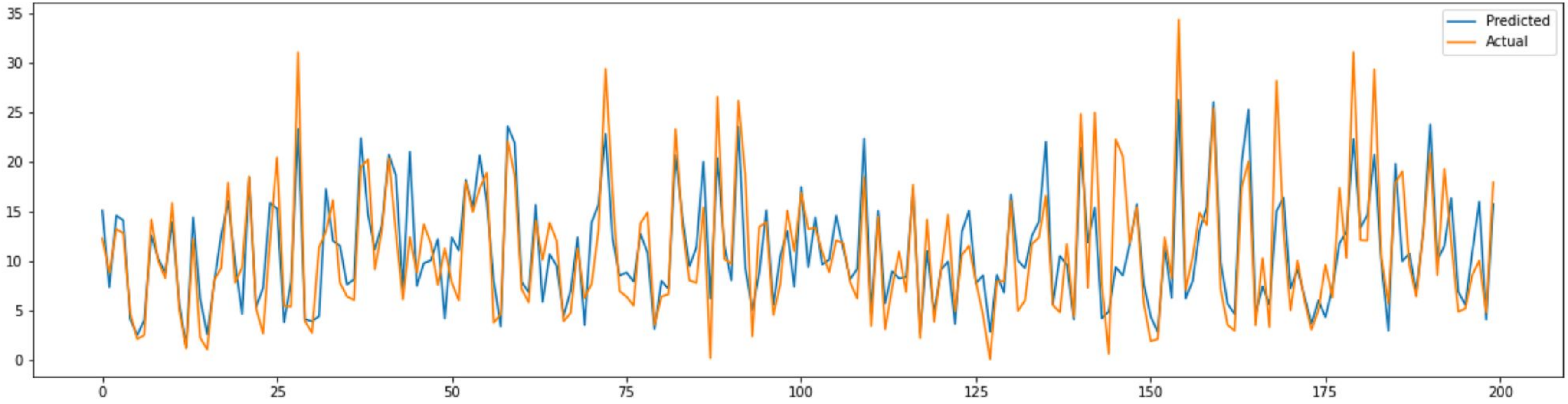
The parameters we will be tuning to get better performances are:
max_depth, learning_rate, min_samples_leaf and max_iter.

5 Cross validations for each set will be conducted in order to find the best parameters.

Hist Gradient Boosting Regressor	RMSE	R2_score
Before Hyperparameter Tuning	3.726250	0.686527
After Hyperparameter Tuning	3.367436	0.743991

Improvement can be seen in the model after the hyperparameter tuning as the RMSE has reduced by 0.358814 and R2_score has increased by 0.057464%.

Actual vs Predicted Values:



Since the data is very large we will consider the first 200 values to compare the actual and predicted trip_durations.
As we can see the model has done a pretty good job in predicting the durations.
Hence it would be safe to say that Hist Gradient Boosting Regressor can be used for future predictions.

Conclusions:

1. Distance calculated using the haversine function plays an important role in predicting the trip durations.

2. Rest of the features showed moderate to very little linear correlation with the dependent variable.

3. The best algorithm in this case is Hist Gradient Boosting Regressor.

4. The untuned model was able to explain 68% of the variance on the test set, while the tuned model explained 74% of variance on the test set which is a good improvement.

5. The least RMSE on test set by the Hist Gradient Boosting Regressor was 3.367436 which is comparatively lower when compared with rest of the models.

6. Hence, Boosting algorithms are by far the best while dealing with large datasets with most of thee features have very little correlation with the dependent feature.

Challenges Faced:

- Huge amount of data had to be dealt with keeping in mind not to loose anything of value.
- Data contained many outliers which had to be removed.
- Data being huge was very time consuming.
- Optimizing the model was very difficult.

Thank you.