Predicting Taxi Trip Duration

Ishita Agarwal
Vellore Institute of Technology, Chennai
Registered Email id: ishita.agl20@gmail.com

Introduction

The NYC Taxi Trip dataset provides a valuable opportunity to explore and predict the total ride duration of taxi trips in New York City. It is released by the **NYC Taxi and Limousine Commission (TLC)** and contains essential information such as pickup time, geographic coordinates, passenger counts, and other relevant variables. The objective of this project is to develop a predictive model that accurately estimates the duration of taxi trips.

Accurately predicting taxi trip duration has practical implications for optimizing transportation logistics and improving passenger experiences. Through the analysis of the NYC Taxi Trip dataset, patterns and factors that influence ride duration will be uncovered, leading to the development of a reliable prediction model.

Insights into the dynamic taxi ecosystem of New York City, where millions of taxi trips occur annually, can be gained from the dataset. The accurate estimation of ride duration is crucial for optimizing dispatching algorithms, estimating travel times, and anticipating traffic congestion.

In this project, **exploratory data analysis** techniques, **data preprocessing methods**, and machine learning modelling techniques majorly **regression** will be employed to extract meaningful insights from the NYC Taxi Trip dataset. The project report will present the findings, methodologies, and outcomes, with a focus on the steps taken in data exploration, preprocessing, feature selection, and model development.

The analysis conducted aims to provide practical value and contribute to the understanding of taxi trip duration prediction.

Data Exploration

During the initial phase of data exploration, a fundamental analysis was conducted to gain an **understanding** of the NYC taxi trip dataset.

A Glimpse of The Data:

	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 dataset comprises a significant volume of data, consisting a total of **1,458,644** data points and featuring **11** distinct attributes. Collectively, these attributes yield valuable insights into the domain of predicting taxi trip durations.

The Attributes:

```
id                   object
vendor_id             int64
pickup_datetime      object
dropoff_datetime     object
passenger_count       int64
pickup_longitude    float64
pickup_latitude     float64
dropoff_longitude   float64
dropoff_latitude    float64
store_and_fwd_flag   object
trip_duration         int64
dtype: object
```

The dataset encompasses attributes that offer insights into NYC taxi trips. The **id** attribute is a unique identifier for each trip, while **vendor_id** indicates the taxi provider. Time information is captured by **pickup_datetime** and **dropoff_datetime**. The **passenger_count** reveals ride occupancy. Geographical coordinates are stored in **pickup_longitude**, **pickup_latitude**, **dropoff_longitude**, and **dropoff_latitude**. The **store_and_fwd_flag** indicates whether data was stored before transmission. Finally, **trip_duration** quantifies trip length. These attributes collectively provide a comprehensive view of taxi travel patterns and characteristics.

The dataset highlights the need for **essential preprocessing steps**. Notably, attributes like **distance and speed are absent**, but they can be computed by leveraging geographical coordinates and trip duration. Furthermore, certain attributes in object format may require **simplification** for smoother analysis. These, concerns are addressed in the next section, i.e. data preprocessing.

Data Description:

	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

The above snapshot describes the data. It presents important statistics like the number of data points, minimum and maximum values, means, standard deviations, and quartile measures. This helps us understand the data's overall behaviour and characteristics quickly and easily.

Evaluating the cleanliness of the data:

An assessment of data cleanliness was conducted, focusing on the identification of duplicate entries and missing values. This examination revealed that **no instances of duplicate records or null values were detected** within the dataset.

Note: *After the initial exploration of the dataset, a need for* ***data preprocessing*** *was identified. Thus, the process of* ***outlier identification and handling*** *was conducted in the next stage, after performing some* ***data preprocessing*** *and introducing additional attributes. Moreover, an extensive* ***exploratory data analysis*** *is carried out on the processed data and documented in the* ***data visualisation*** *section.*

Data Preprocessing

In the previous section, a rationale was established for the necessity of data preprocessing. This essential step took place prior to analysis, aiming to ensure the dataset's reliability, accuracy, and relevance. Notably, significant attributes such as **distance** and **speed** were computed, alongside **formatting** the data appropriately. Furthermore, the identification and management of **outliers** were carried out. These collective efforts laid the groundwork for insightful analysis and the ability to make accurate decisions.

Simplifying Date and Time Attributes:

To make the dataset more manageable and insightful, a process of data simplification was carried out. Date and time information, initially in datetime format, was transformed into easily interpretable attributes. The same is illustrated in the below snapshot.

```
data['week_day'] = data.pickup_datetime.dt.strftime('%A')
data['week_day_num'] = data.pickup_datetime.dt.weekday
data['month'] = data.pickup_datetime.dt.month
data['pickup_hour'] = data.pickup_datetime.dt.hour
```

```
data.head()
```

_count	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	week_day	week_day_num	month	pickup_hour
1	-73.982155	40.767937	-73.964630	40.765602	N	455	Monday	0	3	17
1	-73.980415	40.738564	-73.999481	40.731152	N	663	Sunday	6	6	0
1	-73.979027	40.763939	-74.005333	40.710087	N	2124	Tuesday	1	1	11
1	-74.010040	40.719971	-74.012268	40.706718	N	429	Wednesday	2	4	19
1	-73.973053	40.793209	-73.972923	40.782520	N	435	Saturday	5	3	13

The **week_day** attribute was created by converting the pickup datetime into the corresponding **day of the week** (e.g., Monday, Tuesday), aiding in understanding travel patterns based on the day. The **week_day_num** attribute assigned numeric values to each day (0 for Monday, 6 for Sunday), facilitating quantitative analysis of weekly trends. Similarly, the **month** attribute captured the month of the pickup, enabling insights into **monthly variations**. Additionally, the **pickup_hour** attribute extracted the **hour of the day** from the pickup datetime, enhancing the analysis of hourly travel trends. This transformation step not only simplified the data but also provided key attributes for further exploration and analysis.

Computing Essential Attributes:

In this phase, the extraction of crucial attributes for analysis was undertaken. The process involved calculating essential attributes, such as **distance** and **speed**, using the **geographical coordinates** and **trip duration data**. This computation was essential as these attributes provide critical insights into the dataset.

```
distance = []
for index in data['pickup_latitude'].index:
    distance.append(geodesic((data['pickup_latitude'].iloc[index], data['pickup_longitude'].iloc[index]), (data['dropoff_latitude
data['distance'] = distance
```

```
data.head()
```

ckup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	week_day	week_day_num	month	pickup_hour	distance
-73.982155	40.767937	-73.964630	40.765602	N	455	Monday	0	3	17	1.502172
-73.980415	40.738564	-73.999481	40.731152	N	663	Sunday	6	6	0	1.808660
-73.979027	40.763939	-74.005333	40.710087	N	2124	Tuesday	1	1	11	6.379687
-74.010040	40.719971	-74.012268	40.706718	N	429	Wednesday	2	4	19	1.483632
-73.973053	40.793209	-73.972923	40.782520	N	435	Saturday	5	3	13	1.187038

The **distance** attribute quantifies the length of each trip, allowing us to understand travel patterns. The distance is measured in **kilo meters**.

```
data['speed_m_s'] = (data['distance'] * 1000)/ data['trip_duration']
```

```
data.head()
```

de	pickup_latitude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	week_day	week_day_num	month	pickup_hour	distance	speed_m_s
55	40.767937	-73.964630	40.765602	N	455	Monday	0	3	17	1.502172	3.301477
15	40.738564	-73.999481	40.731152	N	663	Sunday	6	6	0	1.808660	2.727994
27	40.763939	-74.005333	40.710087	N	2124	Tuesday	1	1	11	6.379687	3.003619
40	40.719971	-74.012268	40.706718	N	429	Wednesday	2	4	19	1.483632	3.458351
53	40.793209	-73.972923	40.782520	N	435	Saturday	5	3	13	1.187038	2.728822

```
data['speed_km_hr'] = (data['distance'] * 3600) / data['trip_duration']
```

```
data.head()
```

ude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	week_day	week_day_num	month	pickup_hour	distance	speed_m_s	speed_km_hr
937	-73.964630	40.765602	N	455	Monday	0	3	17	1.502172	3.301477	11.885316
564	-73.999481	40.731152	N	663	Sunday	6	6	0	1.808660	2.727994	9.820778
939	-74.005333	40.710087	N	2124	Tuesday	1	1	11	6.379687	3.003619	10.813029
971	-74.012268	40.706718	N	429	Wednesday	2	4	19	1.483632	3.458351	12.450063
209	-73.972923	40.782520	N	435	Saturday	5	3	13	1.187038	2.728822	9.823760

The attribute, **speed_m_s** measures the speed in **meters per second**, while **speed_km_hr** expresses the speed in **kilo meters per hour**, aiding in a more relatable understanding of the pace of the trip interpretation of travel velocity. This calculation step greatly enhances the dataset's informative value and supports further analysis and decision-making processes.

Outlier Identification:

During the data preprocessing phase, a critical step is to identify and handle outliers within the dataset. Outliers are data points that significantly deviate from the general pattern of the data and can have a substantial impact on analysis and modelling results. In this section, we delve into the process of identifying and addressing outliers in the NYC Taxi Trip Duration Prediction dataset.

A statistical approach is employed, to identify outliers in several numeric columns within the dataset. The **Z-Score method** was utilized to determine how many standard deviations a data point deviates from the mean of its respective column. A threshold value was set to determine if a data point should be classified as an outlier based on its Z-Score. If the absolute value of the Z-Score exceeded this threshold, the data point was flagged as an outlier.

The threshold value of **3** was chosen for identifying outliers based on z-scores in the dataset. This decision was influenced by a **common practice in statistical analysis**, where a z-score above 3 is considered as a strong indication of an extreme outlier. Using this threshold helps in capturing data points that deviate significantly from the mean, allowing us to identify potentially erroneous or anomalous values. The choice of this threshold strikes a balance

between being sensitive enough to identify meaningful outliers and avoiding the exclusion of too many valid data points, ultimately contributing to a more reliable and accurate analysis.

```
z_scores = stats.zscore(data['trip_duration'])
threshold = 3
outliers = data[abs(z_scores) > threshold]
print(outliers.shape)
outliers.head()
```

```
(2073, 18)
```

itude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	week_day	week_day_num	month	pickup_hour	distance	speed_m_s	speed_km_hr
11489	-74.009956	40.714611	N	84594	Saturday	5	2	4	2.988912	0.035332	0.127197
50919	-73.976280	40.750889	N	86149	Saturday	5	5	18	1.179094	0.013687	0.049272
77649	-73.981033	40.743713	N	86352	Tuesday	1	6	12	4.364658	0.050545	0.181962
09217	-73.979584	40.784714	N	86236	Saturday	5	2	0	1.858770	0.021554	0.077596
86992	-73.972336	40.751511	N	85197	Friday	4	3	11	2.145191	0.025179	0.090645

The **trip duration** id is the first attribute taken into consideration while identifying the outliers. The duration of the taxi trips is a central attribute in this project. It's crucial to spot trips that are exceptionally long or short, as these could signify errors or unusual circumstances. Subsequently **distance** and **speed** are also analysed for identifying the outliers.

```
z_scores = stats.zscore(data['distance'])
threshold = 3
outliers = data[abs(z_scores) > threshold]
print(outliers.shape)
outliers.head()
```

```
(40104, 18)
```

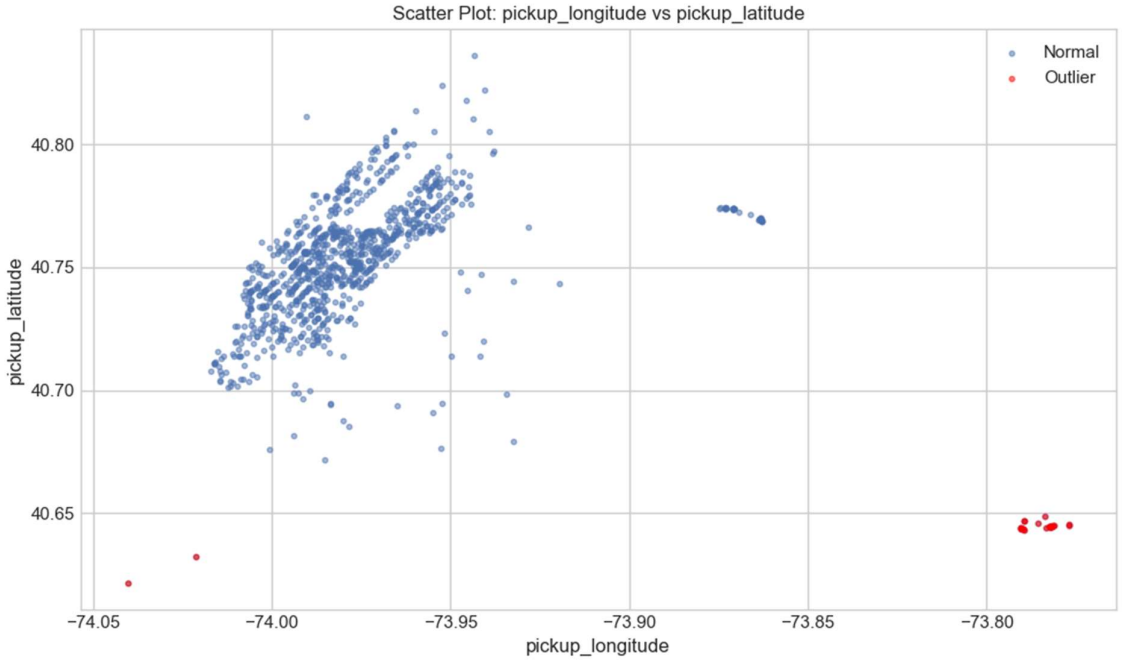
ide	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	week_day	week_day_num	month	pickup_hour	distance	speed_m_s	speed_km_hr
134	-73.788750	40.641472	N	2485	Friday	4	6	8	20.612408	8.294732	29.861034
861	-73.809006	40.816875	N	1557	Tuesday	1	1	23	17.373834	11.158532	40.170715
546	-73.981125	40.720886	N	1782	Wednesday	2	4	23	18.806512	10.553598	37.992953
707	-73.978699	40.750343	N	2065	Friday	4	2	20	19.883300	9.628717	34.663380
160	-73.971771	40.749409	N	1884	Monday	0	6	20	19.611575	10.409541	37.474347

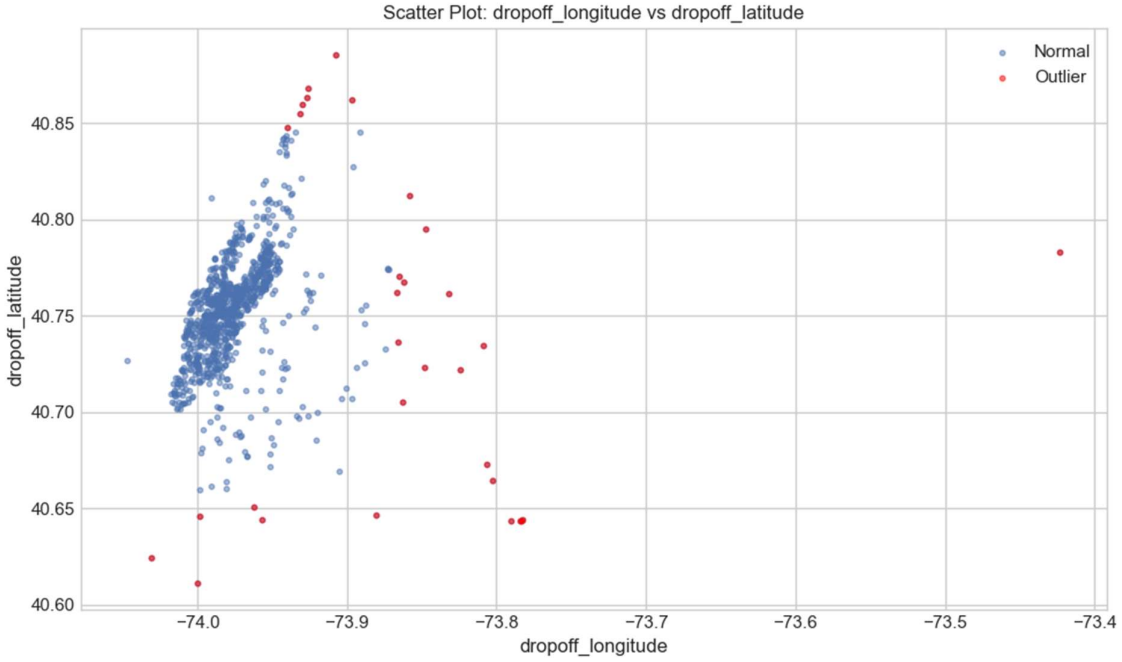
```
z_scores = stats.zscore(data['speed_m_s'])
threshold = 3
outliers = data[abs(z_scores) > threshold]
print(outliers.shape)
outliers.head()
```

```
(737, 18)
```

ide	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	week_day	week_day_num	month	pickup_hour	distance	speed_m_s	speed_km_hr
047	-73.593582	41.043865	N	2534	Wednesday	2	2	0	45.143322	17.815044	64.134159
175	-73.822113	40.711452	N	2	Thursday	3	6	13	0.703342	351.670765	1266.014753
541	-73.935776	40.848473	N	1515	Sunday	6	6	4	26.099120	17.227142	62.017711
314	-73.795242	40.644669	N	7	Wednesday	2	1	20	0.153147	21.878189	78.761479
961	-73.872818	40.774250	N	926	Sunday	6	4	8	15.778745	17.039681	61.342853

Outliers in **latitude** and **longitude** represent unusual geographic coordinates that deviate significantly from the expected range for a given location. Identifying these outliers is important to ensure accurate and reliable geographic data. Removing such outliers enhances the quality of geographical analysis, prevents distorted visualizations, and ensures that data accurately reflects real-world locations.





The presented graphs illustrate a spatial distribution of geographic points. Points marked in red indicate outliers, whereas those in blue represent normal data points.

The **range** of latitudes and longitudes that are classified as outliers is given in the following image.

```
coordinate_columns = ['pickup_longitude', 'pickup_latitude', 'dropoff_longitude', 'dropoff_latitude']
z_threshold = 3
z_scores = {}
for column in coordinate_columns:
    z_scores[column] = stats.zscore(data[column])
coordinate_ranges = {}
for column in coordinate_columns:
    lower_bound = data[column][z_scores[column] <= z_threshold].min()
    upper_bound = data[column][z_scores[column] <= z_threshold].max()
    coordinate_ranges[column] = (lower_bound, upper_bound)
coordinate_ranges
```

```
{'pickup_longitude': (-121.93334197998048, -73.76089477539062),
 'pickup_latitude': (34.35969543457031, 40.84952545166016),
 'dropoff_longitude': (-121.9333038330078, -73.76152038574217),
 'dropoff_latitude': (32.1811408996582, 40.85947036743164)}
```

No significant outliers were detected for key attributes like **month**, **week day**, and **pickup hour.** This suggests that these attributes contain consistent and reasonable data points, without extreme values that could impact analysis or results.

Outlier Removal:

During the data preprocessing phase, the crucial task of identifying and addressing outliers significantly contributes to enhancing data reliability and quality, thereby rendering it more suitable for subsequent analysis and modelling. Notably, in this specific case, the outliers were relatively infrequent in comparison to the overall data size. Consequently, these outlier-laden trip records were systematically removed from the dataset, ensuring a more refined and accurate dataset for further analysis.

```
z_scores = {
    'speed_m_s': stats.zscore(data['speed_m_s']),
    'trip_duration': stats.zscore(data['trip_duration']),
    'distance': stats.zscore(data['distance'])
}
z_threshold = 3
for column, z_scores_array in z_scores.items():
    outliers = data.loc[abs(z_scores_array) > z_threshold]
    data = data.loc[abs(z_scores_array) <= z_threshold]
    print(f"Removed outliers in {column}. New shape: {data.shape}")
coordinate_columns = ['pickup_longitude', 'pickup_latitude', 'dropoff_longitude', 'dropoff_latitude']
coordinate_ranges = {}
for column in coordinate_columns:
    lower_bound = data[column].min()
    upper_bound = data[column].max()
    coordinate_ranges[column] = (lower_bound, upper_bound)
print("Calculated Coordinate Ranges:")
print(coordinate_ranges)
for column, (lower, upper) in coordinate_ranges.items():
    data = data[
        (data[column] >= lower) & (data[column] <= upper)
    ]
```

```
Removed outliers in speed_m_s. New shape: (1457907, 18)
Removed outliers in trip_duration. New shape: (1455834, 18)
Removed outliers in distance. New shape: (1416037, 18)
Calculated Coordinate Ranges:
{'pickup_longitude': (-121.93334197998048, -61.33552932739258), 'pickup_latitude': (34.35969543457031, 43.91176223754882), 'dro
poff_longitude': (-121.9333038330078, -61.33552932739258), 'dropoff_latitude': (34.35969543457031, 43.91176223754882)}
```

The provided screenshot outlines a systematic process utilized to eliminate outliers, underscoring the commitment to dataset integrity. Leveraging Z-scores, significant deviations within various attributes are detected, with a predefined threshold serving as the criterion for outlier identification. The isolation of these outliers from the main dataset helps prevent undue

influence on subsequent analysis, ultimately contributing to the dataset's overall accuracy and reliability.

Exporting the Pre-Processed Data into a File:

The pre-processed data has been saved to a CSV file. To verify its proper export, we can examine the first few rows, the shape, and the data types of the exported data.

```
data.shape
```

```
(1416037, 18)
```

```
data.to_csv('train_preprocessed.csv', index = False)
```

```
preprocessed_data = pd.read_csv('train_preprocessed.csv')
preprocessed_data.head()
```

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

```
preprocessed_data.shape
```

```
(1416037, 18)
```

```
data.dtypes
```

```
id                          object
vendor_id                    int64
pickup_datetime     datetime64[ns]
dropoff_datetime    datetime64[ns]
passenger_count              int64
pickup_longitude           float64
pickup_latitude            float64
dropoff_longitude          float64
dropoff_latitude           float64
store_and_fwd_flag          object
trip_duration                int64
week_day                    object
week_day_num                 int64
month                        int64
pickup_hour                  int64
distance                   float64
speed_m_s                  float64
speed_km_hr                float64
dtype: object
```

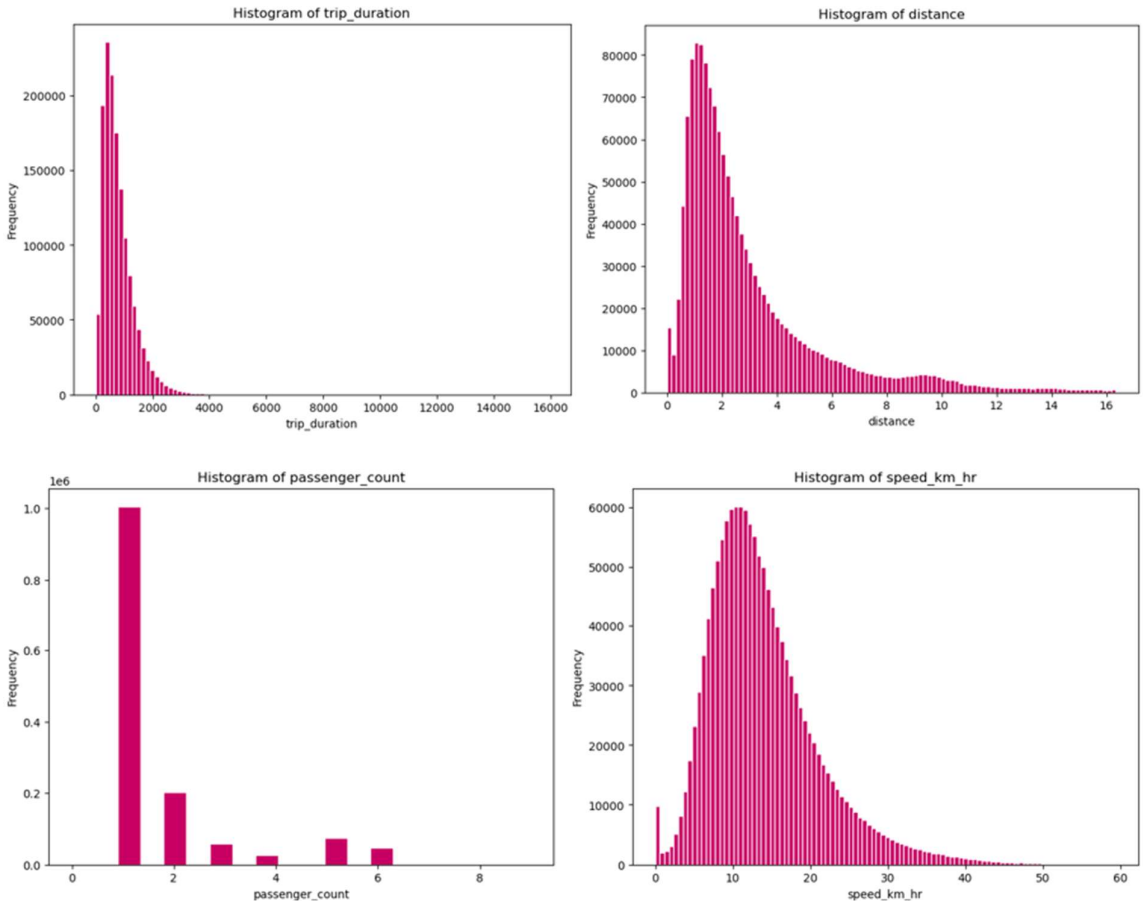
```
data.describe()
```

	vendor_id	passenger_count	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	trip_duration	week_day_num	month	pick
count	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458644e+06	1.458
mean	1.534950e+00	1.664530e+00	-7.397349e+01	4.075092e+01	-7.397342e+01	4.075180e+01	9.594923e+02	3.050375e+00	3.516818e+00	1.360
std	4.987772e-01	1.314242e+00	7.090186e-02	3.288119e-02	7.064327e-02	3.589056e-02	5.237432e+03	1.954039e+00	1.681038e+00	6.399
min	1.000000e+00	0.000000e+00	-1.219333e+02	3.435970e+01	-1.219333e+02	3.218114e+01	1.000000e+00	0.000000e+00	1.000000e+00	0.000
25%	1.000000e+00	1.000000e+00	-7.399187e+01	4.073735e+01	-7.399133e+01	4.073588e+01	3.970000e+02	1.000000e+00	2.000000e+00	9.000
50%	2.000000e+00	1.000000e+00	-7.398174e+01	4.075410e+01	-7.397975e+01	4.075452e+01	6.620000e+02	3.000000e+00	4.000000e+00	1.400
75%	2.000000e+00	2.000000e+00	-7.396733e+01	4.076836e+01	-7.396301e+01	4.076981e+01	1.075000e+03	5.000000e+00	5.000000e+00	1.900
max	2.000000e+00	9.000000e+00	-6.133553e+01	5.188108e+01	-6.133553e+01	4.392103e+01	3.526282e+06	6.000000e+00	6.000000e+00	2.300

Data Visualisation

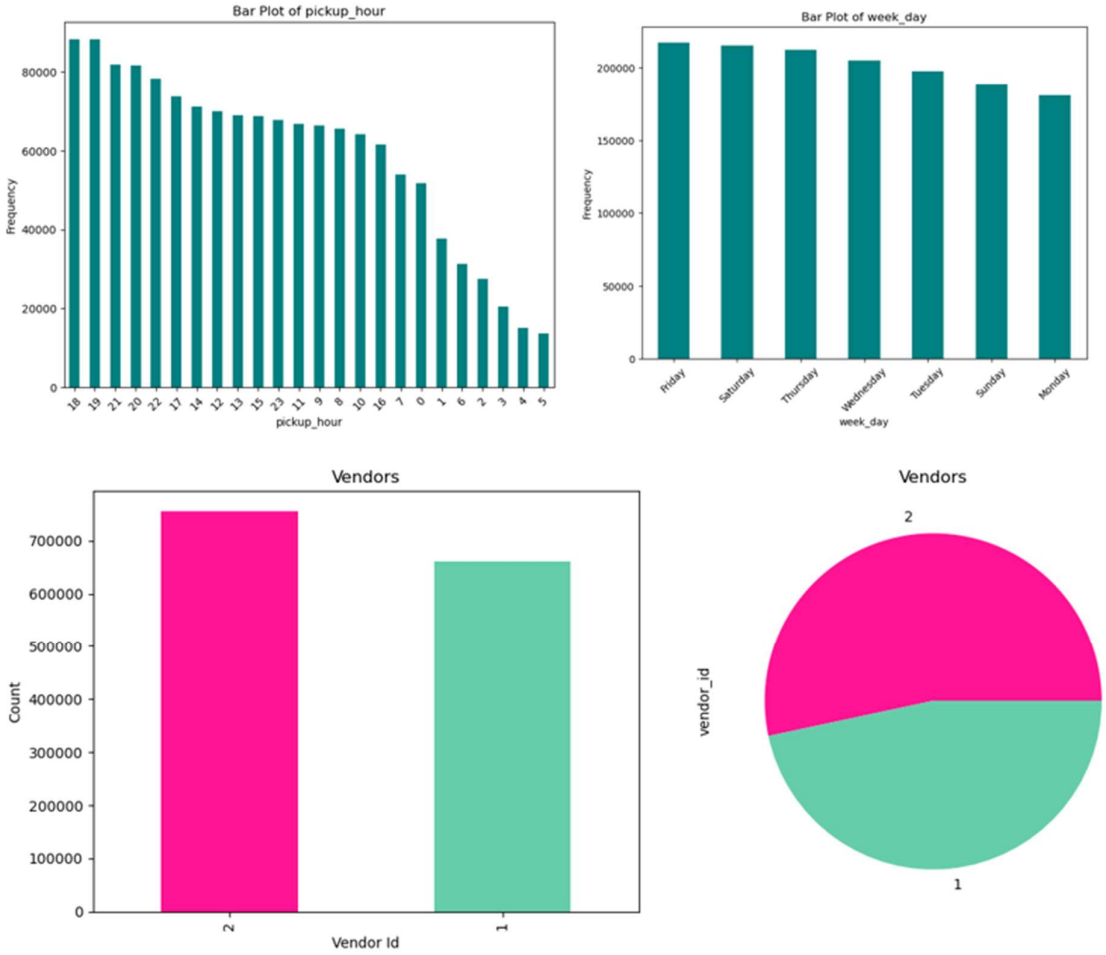
A comprehensive data visualization was performed to **gain deeper insights** into the dataset. Various charts and visualizations were generated to explore the relationships and patterns within the data. Some of the key charts along with the insights derived from them are presented in this section.

Firstly, **univariate analysis** is carried out for individual attributes to understand their distributions and characteristics. This involves generating histograms and bar plots to provide insights into data trends.



The provided graphs offer insights into attribute distributions and their maximum occurrences. The histogram for **trip duration** indicates that the majority of trips lasted under 2000 seconds (about 33 minutes), with a significant number extending between 2000 and 4000 seconds. Similarly, the **distance** histogram illustrates that the majority of trips covered distances below 4 kilometers, while a few extended beyond 16 kilometers. The **speed** graph takes on a bell-shaped form with a positive skew, signifying that most trips had speeds between 10 km/hr and 20 km/hr, and a few even surpassed 40 km/hr. Notably, outliers in these three attributes were eliminated in the prior step. The **passenger count** histogram reveals that most trips involved a

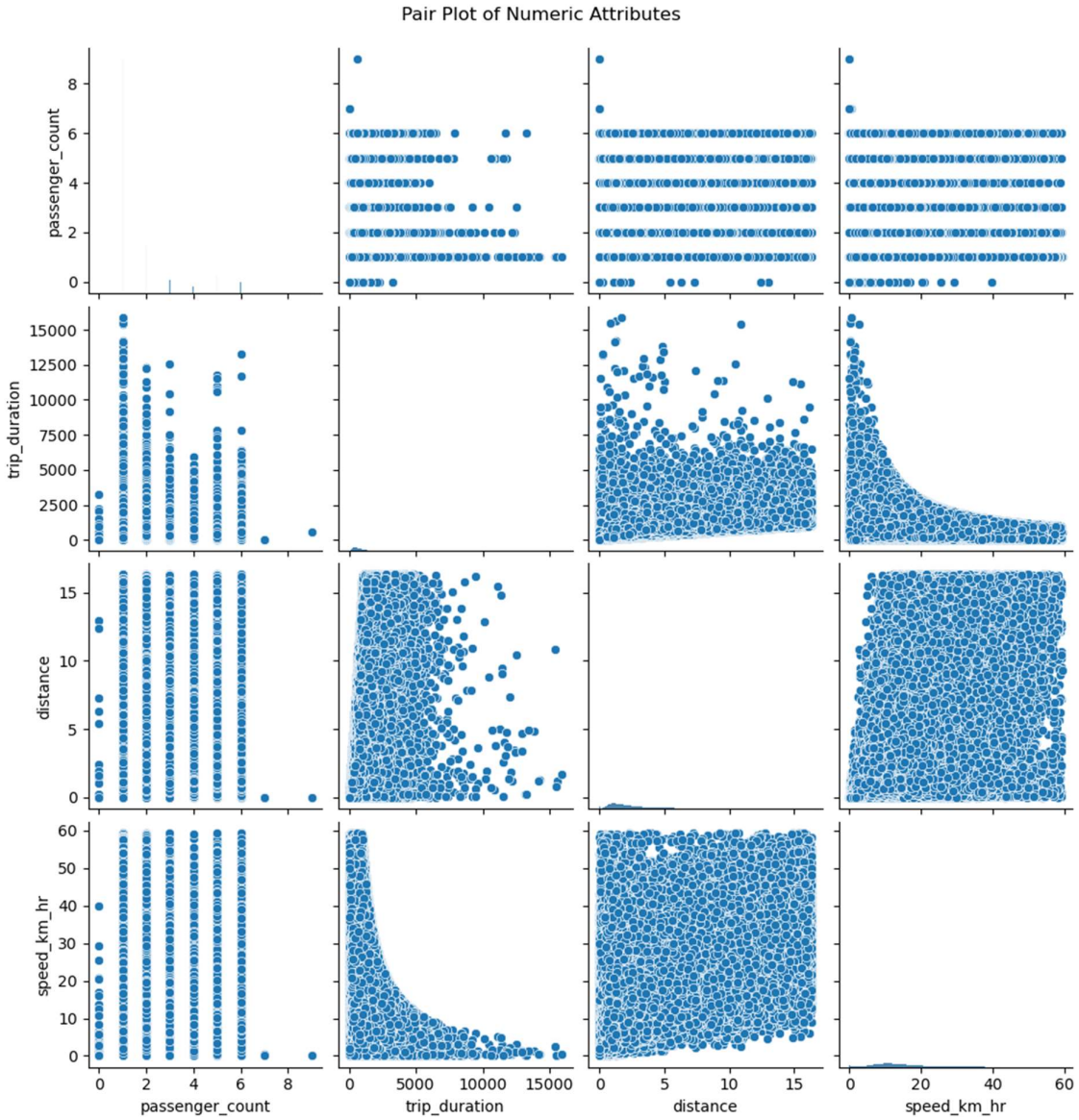
solitary passenger, while a notable proportion included 2 passengers, and the greater values were less frequent.



The presented bar plots provide insights through various visualizations. The **pickup hour** plot reveals that the hours 18 and 19 (6pm and 7pm) have the highest taxi trip frequency, while hour 5 (5am) sees the lowest demand. It's apparent that the hours between 2am and 5am experience the least demand, whereas all other hours are more occupied. Analyzing the **weekday** distribution, an almost equal distribution was observed with all weekdays showing a substantial number of trips, wherein Friday was the busiest and Monday was the least busy day. The **vendor** graphs demonstrate a nearly equal distribution of trips between the two vendors, though vendor 1 has a slight advantage in the dataset

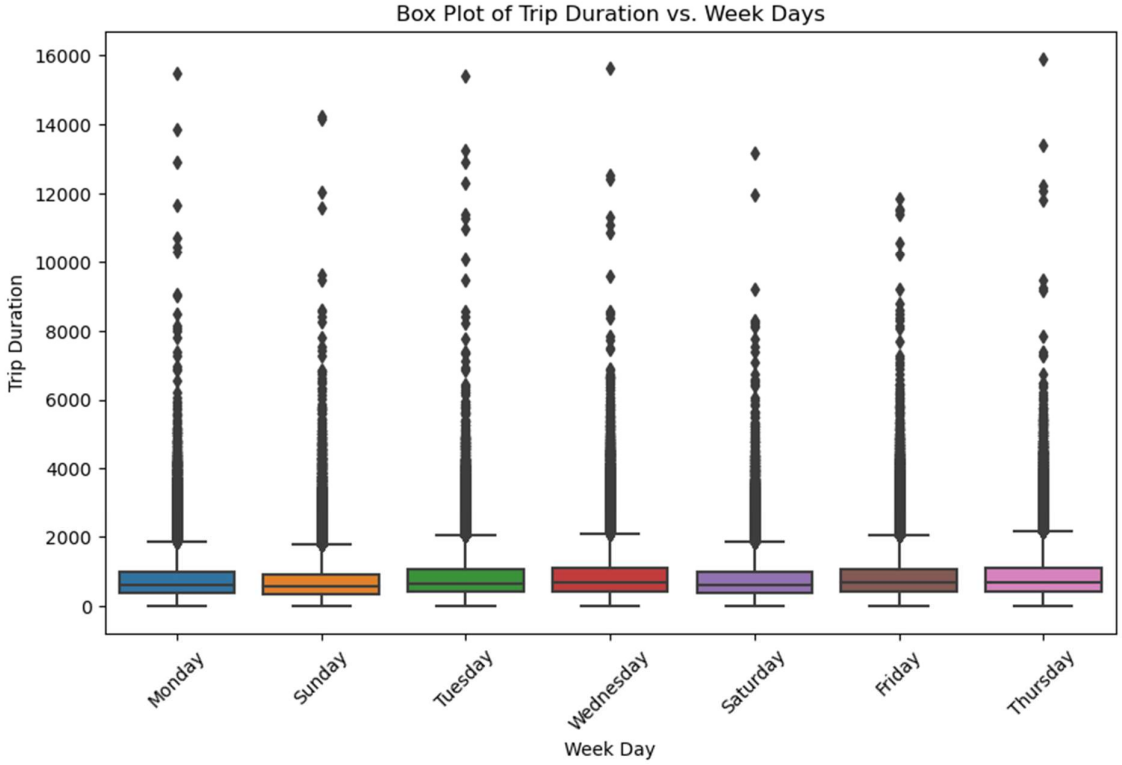
Subsequently, **bivariate analysis** was undertaken to reveal correlations between different attributes. This analysis aids in understanding how changes in one attribute are associated with changes in another. Through a range of visualization techniques, such as scatter plots, box plots, and heatmaps, bivariate analysis facilitates the identification of patterns, trends, and potential dependencies within the dataset. By visually representing these associations, meaningful

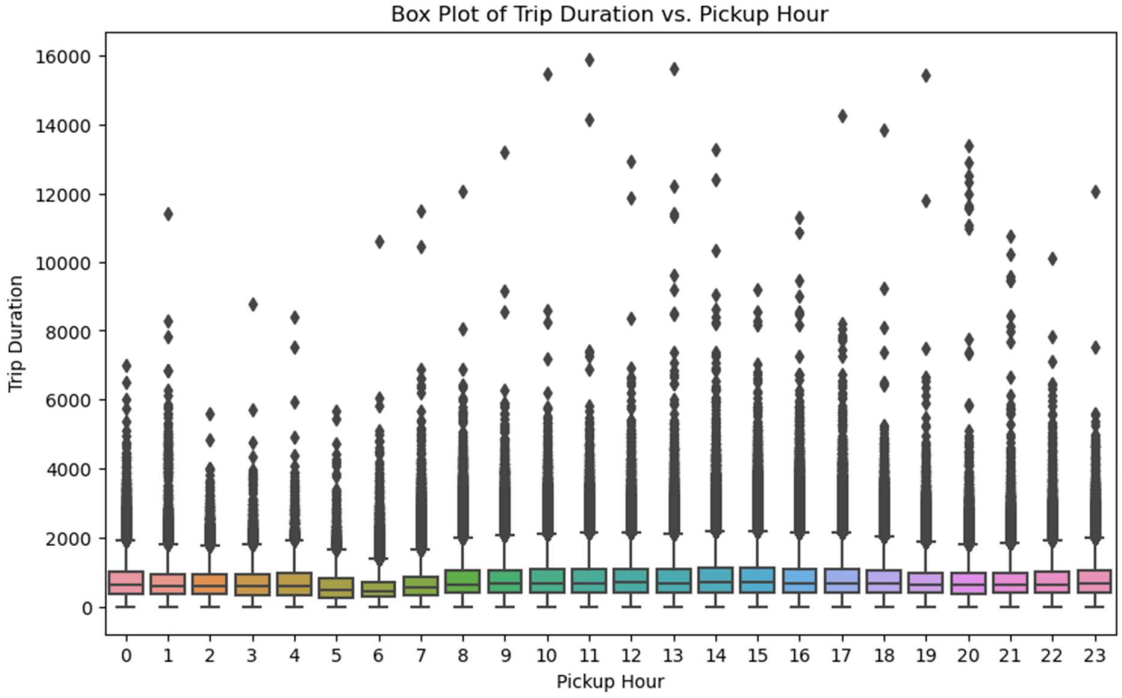
insights can be extracted, contributing to a more comprehensive understanding of the data's underlying dynamics. The document further presents some of these insights along with corresponding graphs.



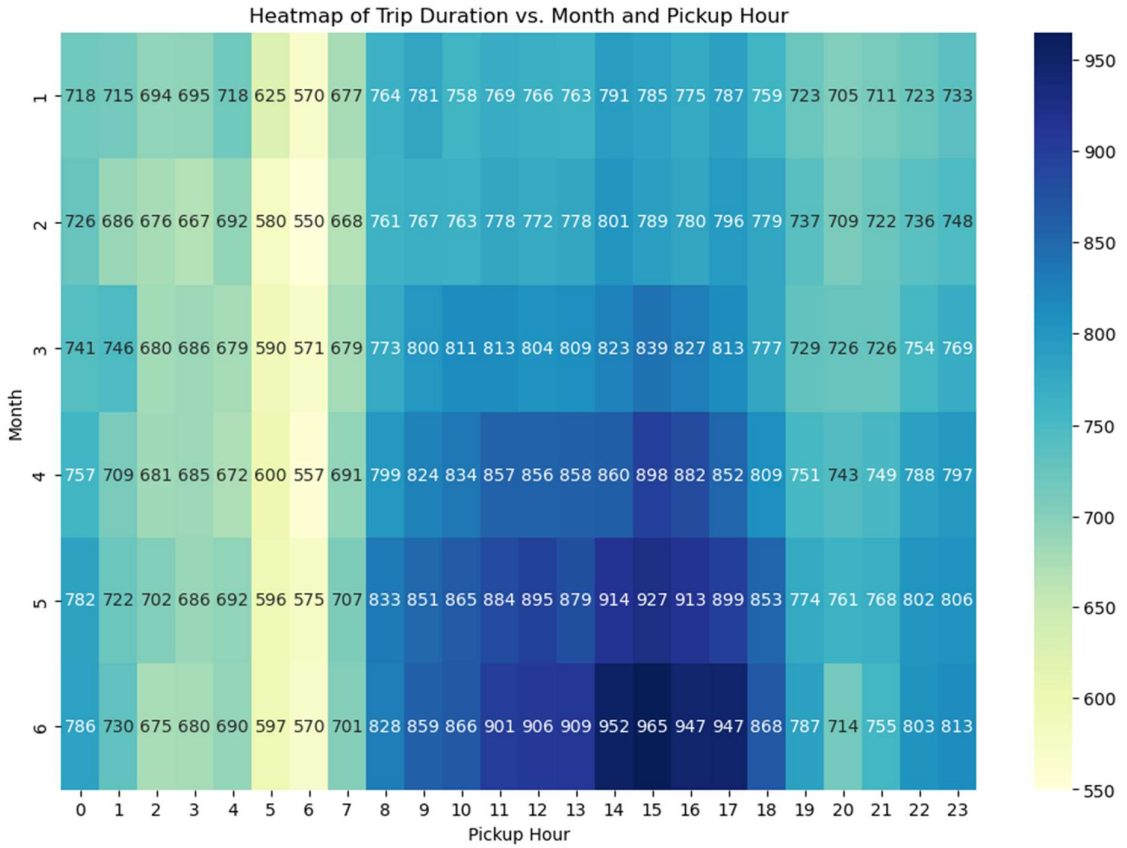
The provided snapshot presents 16 pair plots depicting relationships among four key attributes: passenger_count, trip_duration, distance, and speed. Observing these plots, it becomes evident that **passenger count** has a minimal influence on **trip duration**, while exhibiting no significant impact on distance and speed. No discernible pattern is noticeable between **trip dura**tion and distance. Meanwhile, the anticipated **inverse relationship between trip duration and speed** is observed, indicating that shorter trips tend to have higher speeds, and vice versa. Moreover, the pair plots don't exhibit a straightforward direct relationship between **speed and distance**.

Notably, this proportional trend is more pronounced for **lower speed** values aligned with their corresponding distances, while there is no noticeable trend for higher speed values.

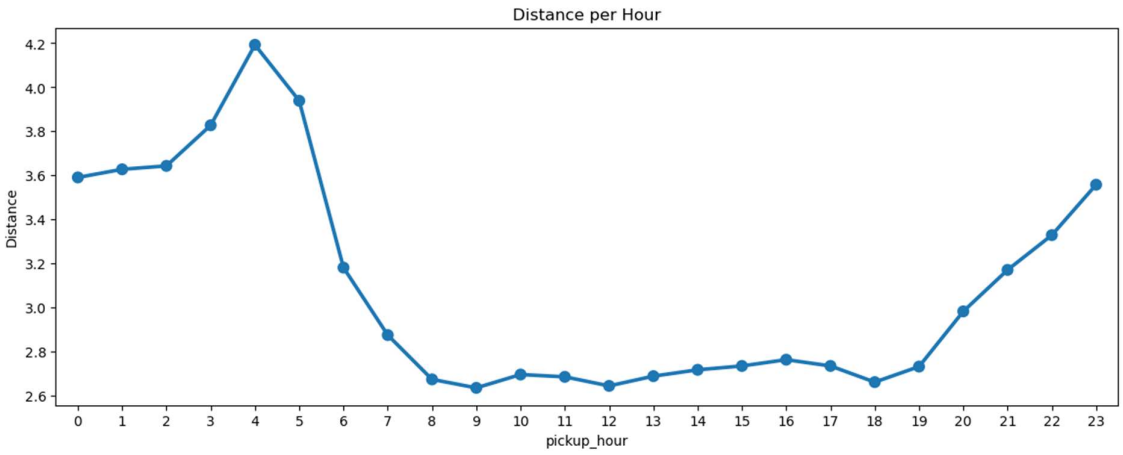




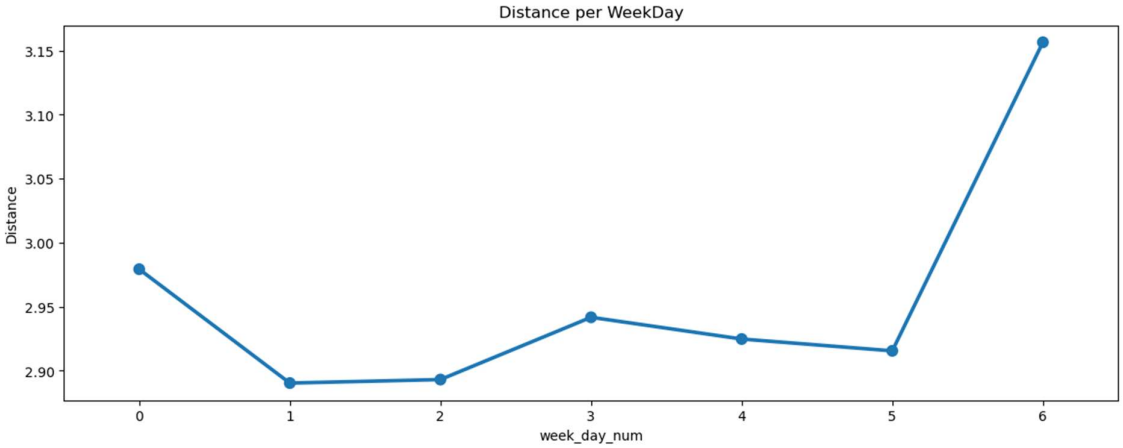
The above Box plots clearly that **week day and pick up hour do not have a very significant affect on the trip duration**. However, it is noticeable that trip during hour 6 (i.e. 6 AM) are of a shorter duration.



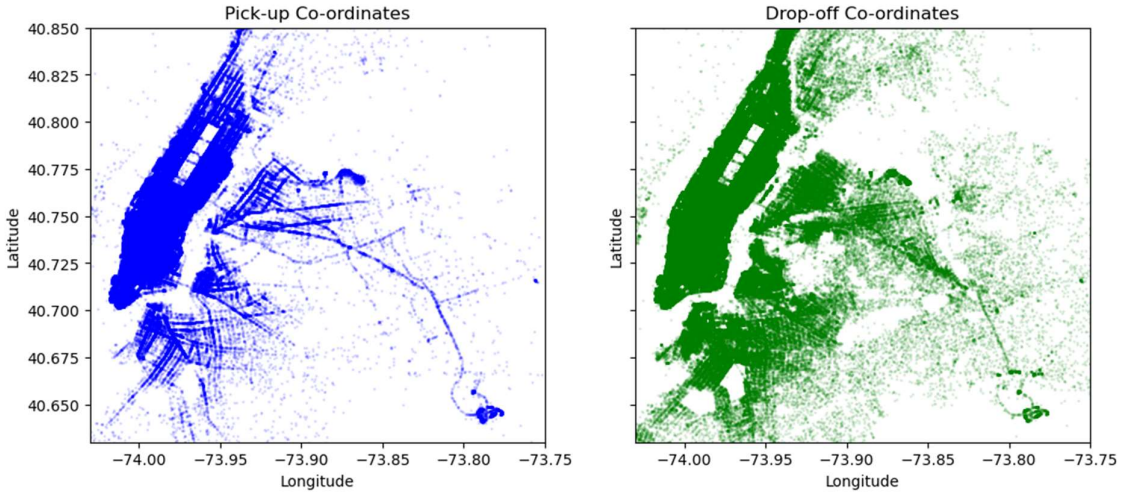
The displayed heatmap showcases the correlation between pickup hours and months. Earlier in the univariate analysis, it was determined that **no distinct bias existed towards any specific hour or month**. However, this heatmap enables the identification of peak hours within each month. Upon careful examination, the heatmap reveals certain trends. For instance, **hour 15 (3:00 PM) and month 6 demonstrate the highest demand**, whereas **hour 6 (6:00 AM) in month 2 exhibits the lowest demand**. Notably, hour 15 consistently remains the peak hour from months 3 to 6, experiencing a nearly 25% surge in demand (between month 1 to month 6). Numerous such insights can be gleaned from this analysis.



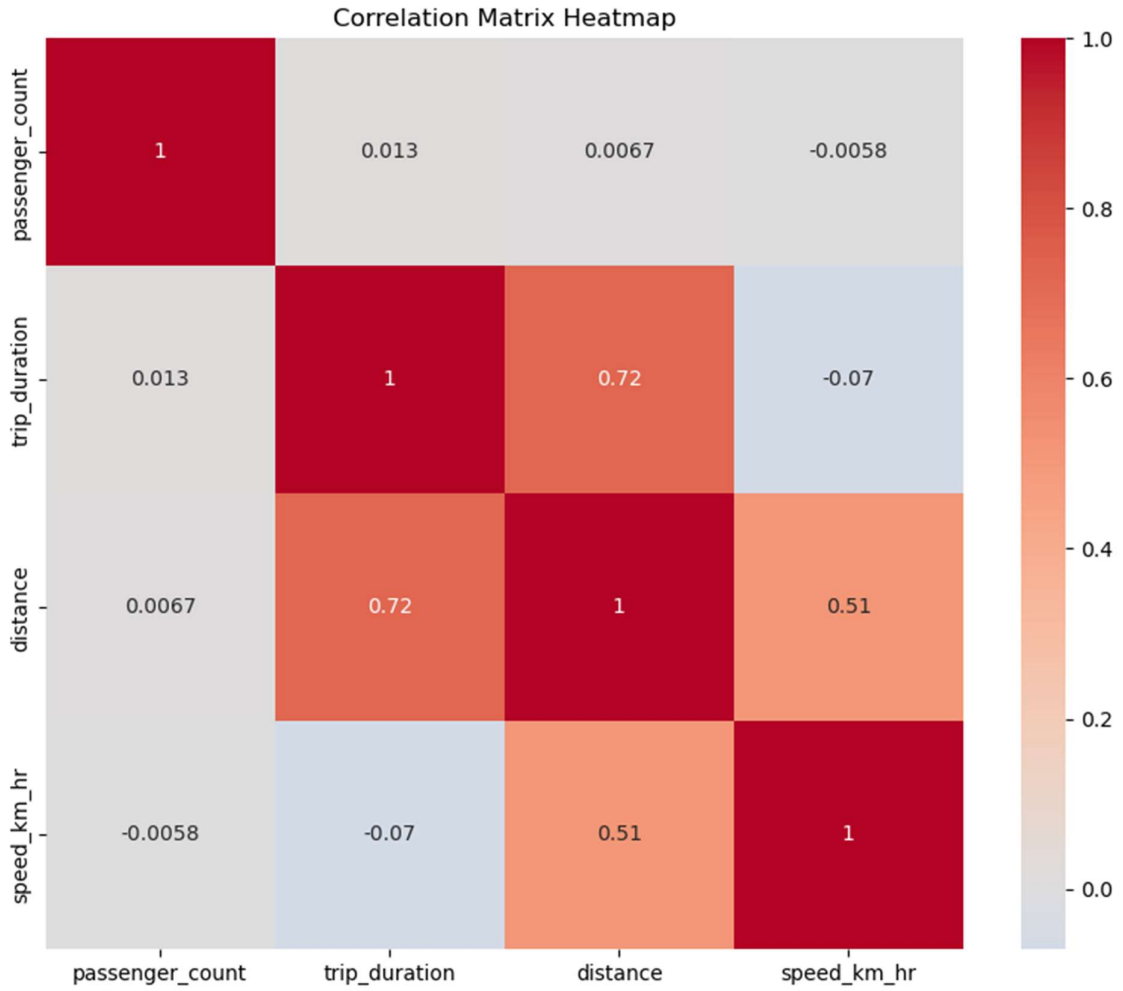
The analysis of trip distance during different hours of the day provides valuable insights. Notably, the trip distance remains relatively **consistent from morning until evening**, fluctuating around 2 to 2.5 kms. However, during the **late-night hours, a gradual increase in trip distance is observed**, starting from the evening and **peaking around 5 AM**, before sharply declining towards the morning hours. This trend may be attributed to various factors such as reduced traffic congestion during these hours or the nature of travel purposes during the late night and early morning periods. Such patterns offer significant implications for understanding travel behaviours and demand during different times of the day.



When analysing the trip distance distribution across different days of the week, it becomes evident that **Saturdays and Sundays** consistently exhibit higher distances compared to the rest of the days. This trend implies that weekends witness longer trips on average, possibly due to leisure activities, travel plans, or reduced traffic congestion. Conversely, weekdays showcase relatively shorter distances, which could be attributed to work-related commutes and routine travel. Overall, the data suggests a distinctive pattern of distance variation between weekends and weekdays.



In the Pickup plot, we can observe that pickups are predominantly **concentrated** in Area X, suggesting a dense demand for rides in that region. On the other hand, drop-off locations exhibit a broader distribution across the city compared to pickups. This trend aligns with the findings from the distance analysis. This widespread distribution of drop-offs within the city, especially in Area X, could be indicative of shorter trips cantered around that specific region.



The correlation heatmap reveals that **passenger count has a minimal influence** on trip duration, whereas **distance and speed exhibit stronger correlations** with trip duration. Notably, distance demonstrates the highest correlation among these attributes, indicating a more significant impact on trip duration.

Data Modelling

In the data modelling section, we transition from exploring and visualizing the dataset to building predictive models that can provide valuable insights and predictions based on the available data. This phase involves **selecting appropriate features**, **preparing the data for training**, and applying various **machine learning algorithms** to create predictive models. By employing techniques such as linear regression, decision trees, ensemble methods, and more, we aim to develop models that can accurately forecast important outcomes or trends within the dataset. This section outlines the steps taken to construct, train, and evaluate these models, showcasing the process of translating data-driven insights into practical predictive capabilities.

Feature Engineering:

The selection of attributes for modelling focuses on those that exhibit meaningful influence on the target variable, trip duration. The chosen attributes include vendor_id, passenger_count, week_day_num, month, pickup_hour, distance, and speed_m_s. These attributes are anticipated to contribute significantly to predicting trip duration. Geographical coordinates, while not directly tied to trip duration, may be subjected to preprocessing such as clustering to extract relevant insights. Notably, the store and forward flag, being a system indicator, is omitted from the modelling process due to its limited expected impact on predicting trip duration.

Data Preparation:

The dataset has been partitioned into two subsets: training and testing. The training set comprises 80% of the data, while the testing set contains the remaining 20%. The parameters for training and testing are meticulously defined based on the selected features from the preceding feature selection step. This division allows for a robust evaluation of the predictive models.

Note: *The testing data provided doesn't have the trip duration column hence it is not possible to evaluate the accuracy of the prediction. Therefore, it is required to split the train data set itself.*

Prediction and Testing:

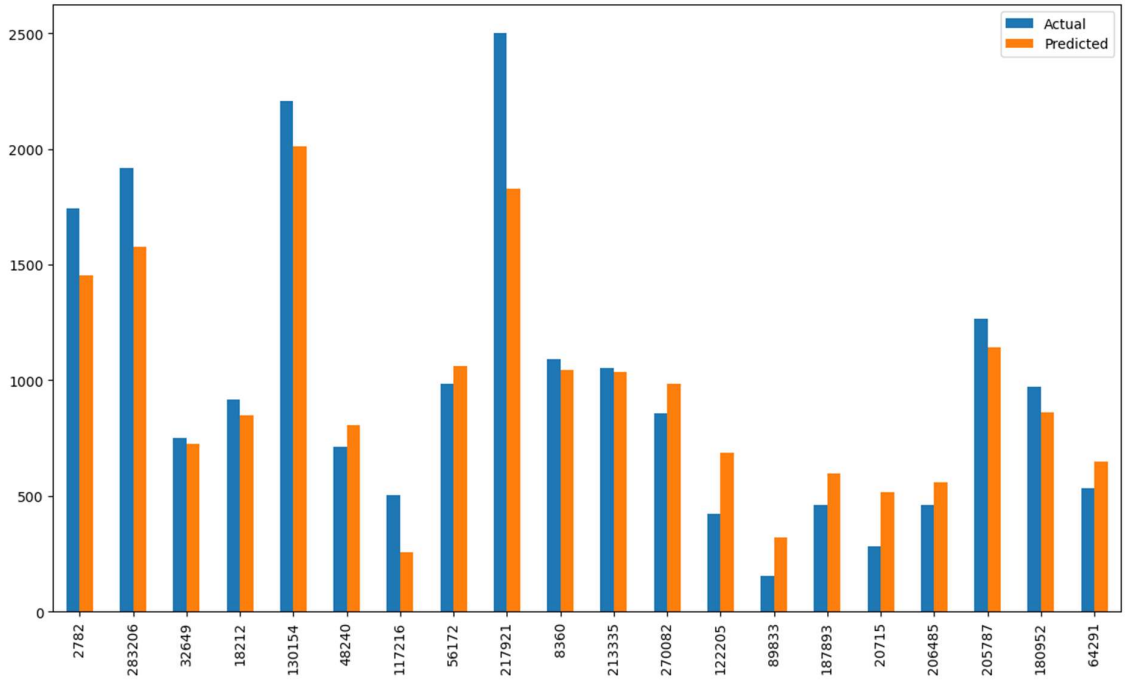
In the pursuit of achieving accurate predictions, a comprehensive evaluation of various algorithms was undertaken. This involved the utilization of well-established techniques such as Linear Regression, Decision Tree, Gradient Boosting, XGBoost, and Random Forest. Each of these algorithms trained and tested on the dataset to gauge their efficacy in predicting trip durations.

The following document states the findings of this evaluation, offering a detailed overview of the accuracy metrics attained by each algorithm. Additionally, the time taken for training these models is meticulously documented, shedding light on their computational efficiency.

Multiple Linear Regression

Incorporating the concept of multiple linear regression, the analysis delved into examining the collective impact of **multiple predictor variables** on the **target variable**, trip duration. The predictor variables, namely vendor_id, passenger_count, week_day_num, month, pickup_hour, distance, and speed_m_s, were carefully selected based on their anticipated influence on trip duration. Through this approach, the interplay of these variables in tandem was explored, aiming to discern their joint effect on the predictive accuracy of the model.

The forthcoming section expounds upon the outcomes of this multiple linear regression analysis, elucidating the degree of accuracy achieved and the implications of these findings in the context of predicting taxi trip durations.



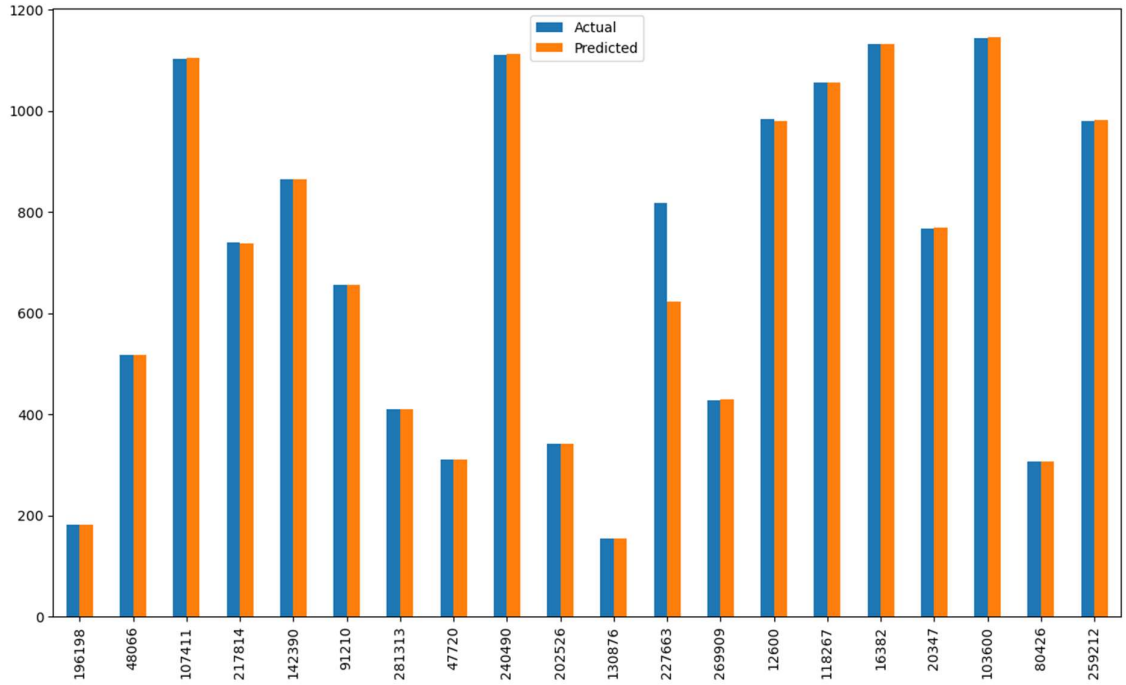
Time taken to train linear regression model: **0.38 seconds**

The graph shows quite some difference between the predicted values and the actual values. The score (coefficient of determination) of this model is **0.7820769321362124**.

Decision Tree

Employing the decision tree algorithm, the analysis embarked on a more intricate exploration of the dataset's attributes and their relationships. The decision tree algorithm sought to capture **non-linear patterns and interactions within the data**, potentially offering a more nuanced understanding of how the predictor variables collectively impact the target variable, trip duration. By iteratively partitioning the data into subsets based on attribute values, the decision tree aimed to discern the most influential attributes for predicting trip durations.

The subsequent section elaborates on the insights gleaned from the decision tree algorithm, detailing the accuracy outcomes and shedding light on the interpretability of the model's outcomes in the context of taxi trip duration prediction.



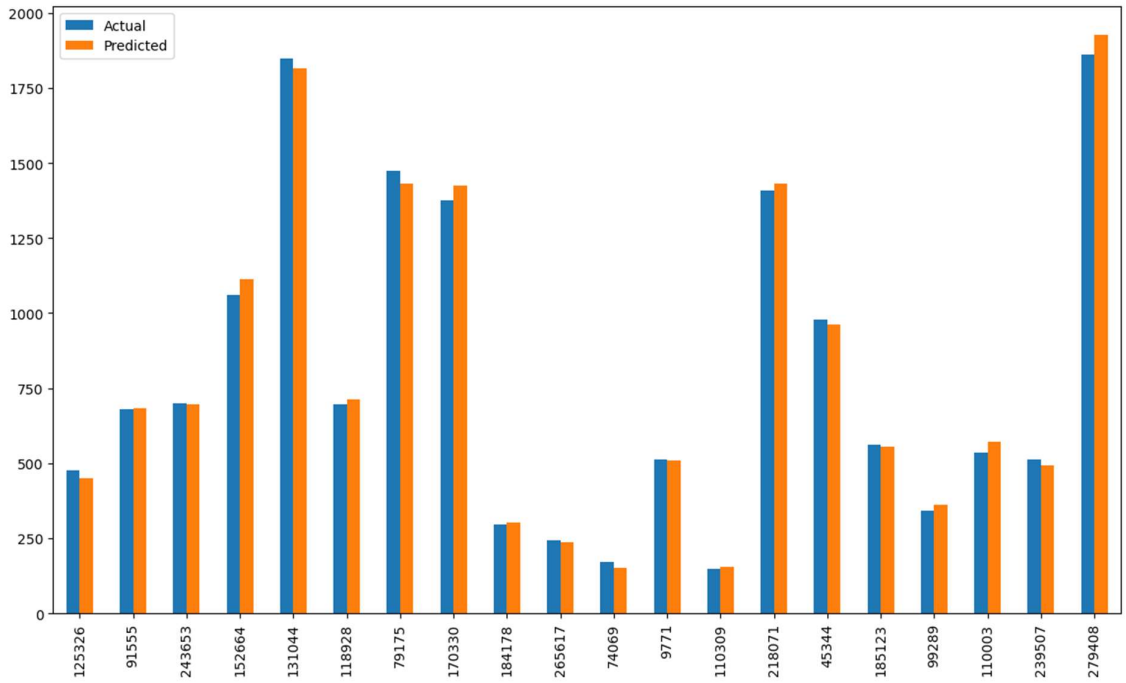
Time taken to train Decision tree model: **14.58 seconds**

The graph shows almost no difference between the predicted values and the actual values for most values except for one instance of data. The score (coefficient of determination) of this model is **0.9931072584176449**.

Gradient Boost

The journey of model exploration continued with Gradient Boosting, a sophisticated machine learning technique that constructs an **ensemble of weak predictive models**, typically decision trees, and sequentially refines them to create a strong predictive model. Gradient Boosting is adept at tackling regression and classification tasks alike, making it a valuable tool in predictive modeling.

In the context of predicting taxi trip durations, Gradient Boosting was harnessed to uncover patterns within the data and generate accurate predictions. This section delves into the application of Gradient Boosting for trip duration prediction, offering insights into its performance metrics and highlighting its capacity to capture intricate relationships within the dataset. The iterative nature of gradient boosting, where subsequent models address the errors of their predecessors, showcases its adaptability to complex prediction tasks.



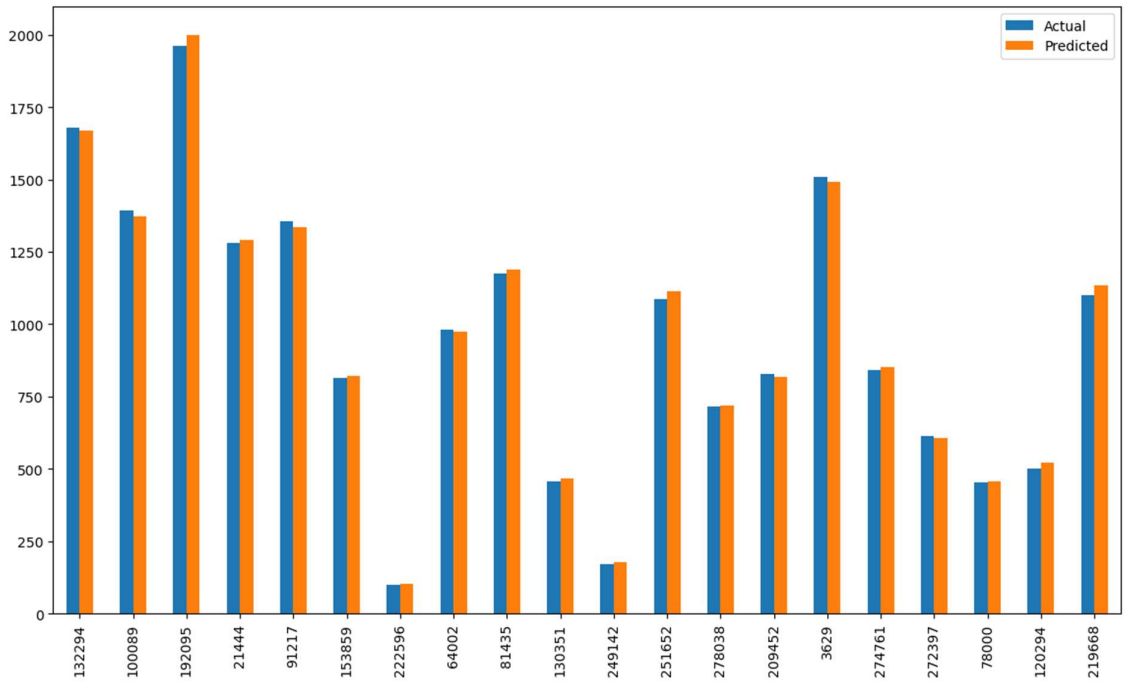
Time taken to train Gradient Boost model: **341.81 seconds**

The graph shows very slight difference between the predicted values and the actual values. The score (coefficient of determination) of this model is **0.9861719142174483**.

XG Boost

Moving forward, the exploration ventured into the realm of XGBoost (Extreme Gradient Boosting), a powerful and efficient gradient boosting algorithm. XGBoost excels at handling both regression and classification tasks by **sequentially building multiple weak learners**, usually decision trees, and iteratively refining them to minimize prediction errors. Its unique optimization techniques, such as gradient boosting and regularization, enhance model performance while avoiding overfitting.

In the context of taxi trip duration prediction, XGBoost was employed to harness its predictive capabilities. This section sheds light on the outcomes of utilizing XGBoost to predict trip durations accurately. The discussion covers the model's performance metrics, emphasizing its ability to capture intricate patterns within the data and deliver robust predictions. The amalgamation of gradient boosting and regularization techniques underscores the model's effectiveness in addressing complex real-world scenarios.



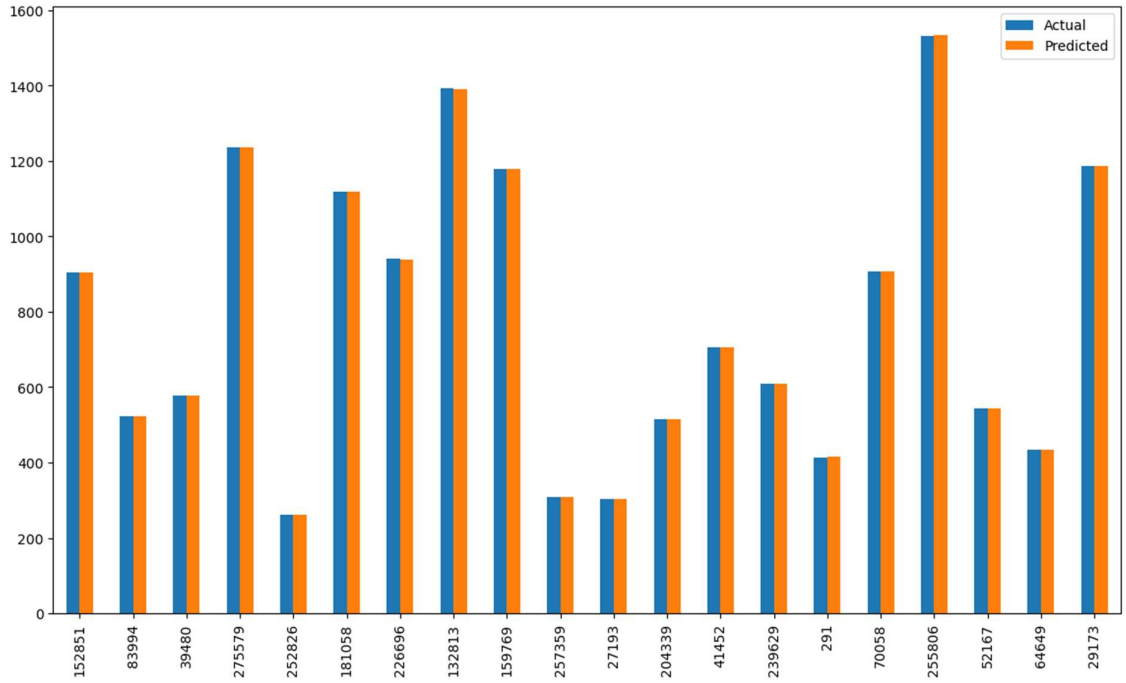
Time taken to train XG Boost model: **77.82 seconds**

The graph shows very slight difference between the predicted values and the actual values. The score (coefficient of determination) of this model is **0.9953883245761678**.

Random Forest

The analysis progressed to utilizing the Random Forest algorithm, a robust **ensemble learning technique** that **combines the power of multiple decision trees** to enhance predictive accuracy. This algorithm harnesses the diversity of numerous decision trees by averaging their predictions, thereby mitigating the potential for overfitting and increasing the model's generalizability. Random Forest operates by generating several decision trees through bootstrapped samples of the data and random subsets of features, allowing for more comprehensive exploration of the data's intricacies.

The subsequent section delves into the outcomes of employing the Random Forest algorithm for taxi trip duration prediction. It presents insights into the model's accuracy, highlighting how the aggregation of multiple decision trees contributes to improved predictive performance and the ability to capture complex relationships within the data.

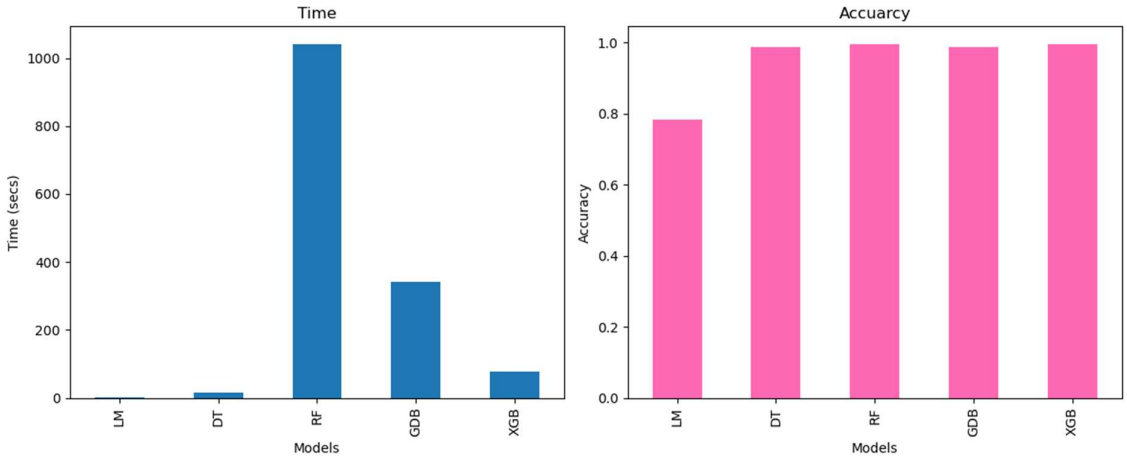


Time taken to train Random Forest model: **77.82 seconds**

The graph shows almost no difference between the predicted values and the actual values. The score (coefficient of determination) of this model **is 0.9950334651209664**.

Comparing the Models:

In evaluating various regression algorithms – Linear Regression, Decision Tree, Gradient Boosting, XGBoost, and Random Forest – for predicting taxi trip durations, it is evident that each algorithm has distinct strengths. The following graphs demonstrate this.



The insights drawn from the comparison of regression algorithms are as follows:

- **Multiple Linear Regression** demonstrated the shortest execution time among the algorithms, but its accuracy was relatively lower.
- **Decision Tree** exhibited faster execution compared to Gradient Boost, XGBoost, and Random Forest, while maintaining a remarkably high level of accuracy.
- **XGBoost** showcased both high accuracy and efficient execution, taking around a minute to train, outperforming Gradient Boost and Random Forest in terms of time.
- **Gradient Boost** and **Random Forest** displayed commendable accuracy, albeit at the expense of efficiency, as their training times were notably longer compared to other algorithms.

In light of the analysis, the **Decision Tree** algorithm emerges as a standout choice, excelling in both accuracy and computational efficiency. Furthermore, **XGBoost** proves to be another strong contender, boasting high accuracy levels and efficient training times. Despite its slightly longer training duration compared to Decision Tree, its exceptional predictive capabilities make it a valuable asset for accurate trip duration prediction.

Conclusion

In this project, we embarked on a journey to predict taxi trip durations in the bustling city of New York. Through the meticulous exploration, analysis, and modelling of the **NYC Taxi Trip dataset**, we have gained valuable insights into the factors that influence ride durations and successfully developed predictive models to estimate trip lengths accurately. This endeavour holds significant implications for optimizing transportation logistics, enhancing passenger experiences, and improving urban mobility.

Our exploration of the dataset unveiled the intricate dynamics of the taxi ecosystem. We delved into attributes such as pickup time, passenger counts, geographical coordinates, and more, unravelling patterns that shape taxi travel behaviour. The **data preprocessing phase** equipped us with essential tools to ensure data integrity, addressing outliers, and constructing meaningful attributes like distance and speed. This process fortified the dataset for robust analysis and modelling.

Through **data visualization**, we visualized the relationships within the data, revealing nuanced patterns that aid in understanding travel trends. We identified peak hours, explored weekday variations, and correlated attributes to provide a comprehensive picture of taxi trip behaviours. This visualization stage not only enhanced our grasp of the data but also laid the foundation for the subsequent modelling phase.

The data modelling phase was marked by the selection of influential attributes, data preparation, and the application of various machine learning algorithms. Our predictive journey encompassed algorithms such as Multiple Linear Regression, Decision Tree, Gradient Boosting, XGBoost, and Random Forest. Each algorithm was trained and tested on the dataset, yielding insights into their accuracy and computational efficiency.

In our comparison of these algorithms, two standout performers emerged. The **Decision Tree** algorithm demonstrated exceptional accuracy and efficiency, making it a robust choice for predicting trip durations. Meanwhile, **XGBoost** showcased a harmonious balance of accuracy and training speed, reinforcing its credibility as a potent predictive tool. Moreover, the **prediction was performed on the testing dataset** using the Decision Tree algorithm, reaffirming the practical applicability of our approach.

```
test_data.head()
```

gitude	pickup_latitude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	week_day	week_day_num	month	pickup_hour	distance	predicted_trip_duration
)88129	40.732029	-73.990173	40.756680	N	Thursday	3	6	23	2.742863	403.0
)64203	40.679993	-73.959808	40.655403	N	Thursday	3	6	23	2.755774	539.0
)97437	40.737583	-73.986160	40.729523	N	Thursday	3	6	23	1.307112	275.0
)56070	40.771900	-73.986427	40.730469	N	Thursday	3	6	23	5.266978	796.0
)70215	40.761475	-73.961510	40.755890	N	Thursday	3	6	23	0.961745	352.0

As we conclude this endeavour, we recognize the profound impact of predictive modelling on optimizing urban mobility. Accurate predictions of taxi trip durations have the potential to revolutionize transportation logistics, minimize waiting times, and enhance the overall passenger experience. Our journey through data exploration, preprocessing, visualization, and modelling underscores the power of data-driven insights in addressing real-world challenges.

In essence, this project showcases the fusion of data science methodologies and domain knowledge, illustrating their ability to unravel complex urban dynamics. By predicting taxi trip durations with precision, we contribute to a more efficient and seamless urban transportation landscape, paving the way for smarter cities and enhanced mobility experiences.

This project was done as a part of a simulated summer internship with capabl.

Attachment

Source Code can be found at: https://github.com/ishitaagl20/NYC-Taxi_Trip_Prediction

****