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()
```

ickup_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()
```

tude	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
7937	-73.964630	40.765602	N	455	Monday	0	3	17	1.502172	3.301477	11.885316
3564	-73.999481	40.731152	N	663	Sunday	6	6	0	1.808660	2.727994	9.820778
3939	-74.005333	40.710087	N	2124	Tuesday	1	1	11	6.379687	3.003619	10.813029
9971	-74.012268	40.706718	N	429	Wednesday	2	4	19	1.483632	3.458351	12.450063
3209	-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
1489	-74.009956	40.714611	N	84594	Saturday	5	2	4	2.988912	0.035332	0.127197
0919	-73.976280	40.750889	N	86149	Saturday	5	5	18	1.179094	0.013687	0.049272
7649	-73.981033	40.743713	N	86352	Tuesday	1	6	12	4.364658	0.050545	0.181962
9217	-73.979584	40.784714	N	86236	Saturday	5	2	0	1.858770	0.021554	0.077596
6992	-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)
```

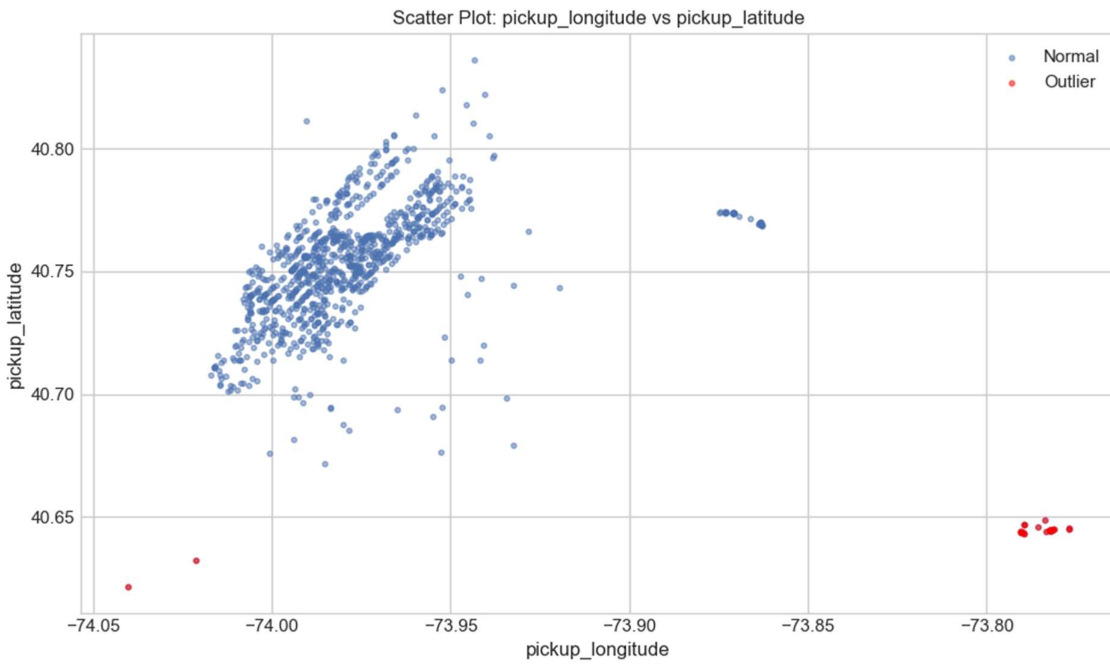
ide	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	week_day	week_day_num	month	pickup_hour	distance	speed_m_s	speed_km_hr
34	-73.788750	40.641472	N	2485	Friday	4	6	8	20.612408	8.294732	29.861034
61	-73.809006	40.816875	N	1557	Tuesday	1	1	23	17.373834	11.158532	40.170715
46	-73.981125	40.720886	N	1782	Wednesday	2	4	23	18.806512	10.553598	37.992953
07	-73.978699	40.750343	N	2065	Friday	4	2	20	19.883300	9.628717	34.663380
60	-73.971771	40.749409	N	1884	Monday	0	6	20	19.611575	10.409541	37.474347

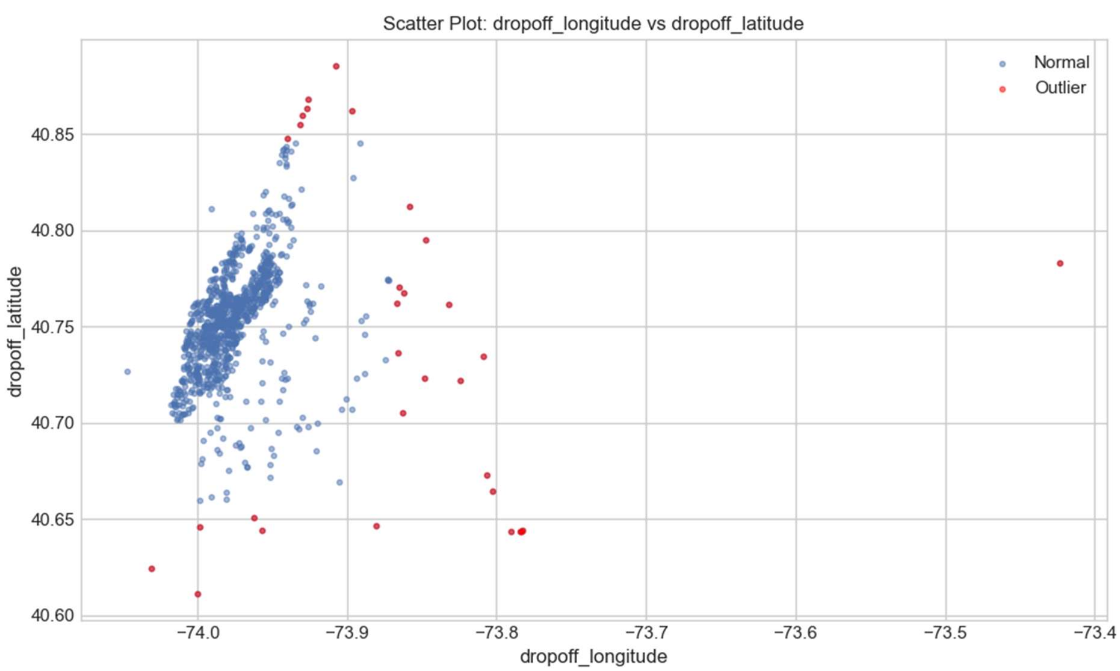
```
z_scores = stats.zscore(data['speed_m_s'])
threshold = 3
outliers = data[abs(z_scores) > threshold]
print(outliers.shape)
outliers.head()
```

```
(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
47	-73.593582	41.043865	N	2534	Wednesday	2	2	0	45.143322	17.815044	64.134159
75	-73.822113	40.711452	N	2	Thursday	3	6	13	0.703342	351.670765	1266.014753
41	-73.935776	40.848473	N	1515	Sunday	6	6	4	26.099120	17.227142	62.017711
14	-73.795242	40.644669	N	7	Wednesday	2	1	20	0.153147	21.878189	78.761479
61	-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