Uber Pickup Data Analysis

Business Requirements

- The goal is to Gather data regarding pickup and drop locations, pick up and drop times, travel times, and fares.
- Analysis of these Data attributes require determine certain things
- Like – how many rides takes per Day, which Area is most crowded , how much Each ride cost.

Project Objectives

- Determine the rush depending on latitude and longitude
- Analysis of journey by weekdays
- Analyzing which month has maximum rides
- Analysis of journey by hours
- Fare of the drives

Literature Review

- Uber Technologies, Inc. was founded in 2009 with the aim of improving the efficiency of taxi services in major cities throughout the United States. As of 2010, the estimated value of the company surpassed 70 billion dollars, making it the world's most valuable privately-owned technology company.

- According to Uber has about **111 million subscribers** and completes about **19 million journeys a day.**

- India has approximately **8 million users**. Uber is at the height of data regarding consumers, thanks to the abundance of data available.

Data Requirements

It is necessary to gather data that has certain characteristics and can assist us in building a statistical model to analyze the pickup and drop times and locations of riders in order to attain the above goal. Additionally, we have considered the fare, the location of the drivers, and the time of the ride.

● Pickup point and time ● Drop Point and time ● Time of travel ● Fare of travel ● Placement of the drivers

Dataset

Uber Price Prediction Datasets

Features

'City’,
'Product Type’,
'Trip or Order Status’,
'Request Time’,
'Begin Trip Time’,
'Begin Trip Lat’
'Begin Trip Lng’,
'Dropoff Time’,
'Dropoff Lat’,
'Dropoff Lng’,
'Distance (miles)’

Target Variable: 'Fare Amount'

Dataset

Uber Pickup Analysis

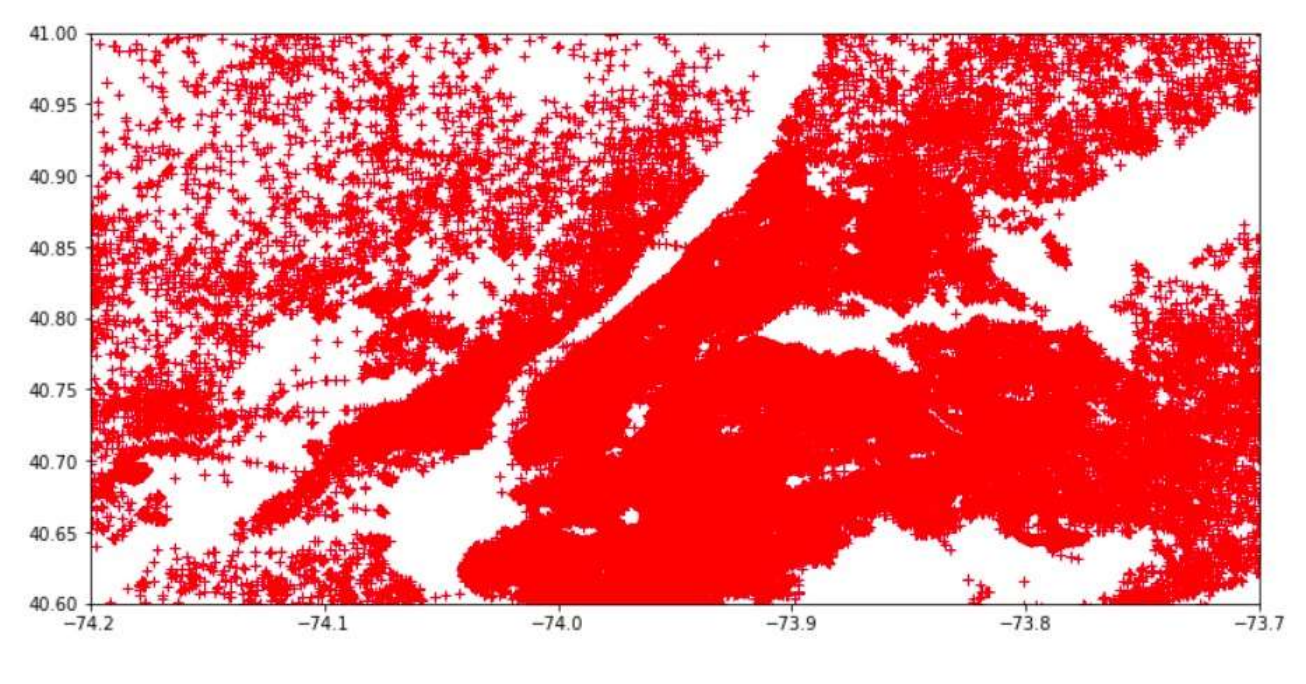
- Features
- Date/Time
- Lat
- Base

FiveThirtyEight’s official site for the data is:uber-tlc-foil-response/uber-raw-data-apr14.csv at master · fivethirtyeight/uber-tlc-foil-response · GitHub

Data Preprocessing

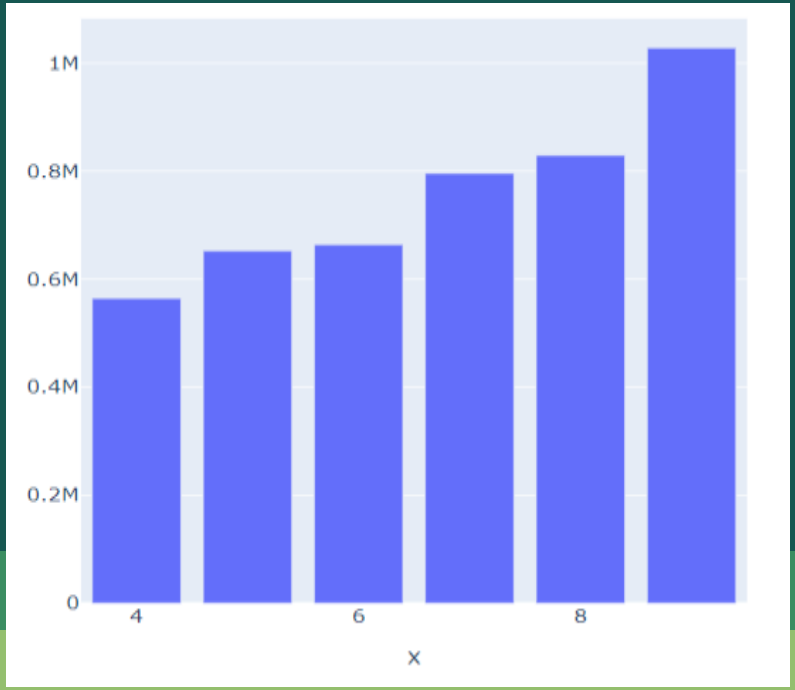
- Examination of data distribution
- Handling missing values of the dataset (a most common issue with every dataset)
- Splitting data with additional columns

Determine the Rush depending on Latitude and Longitude

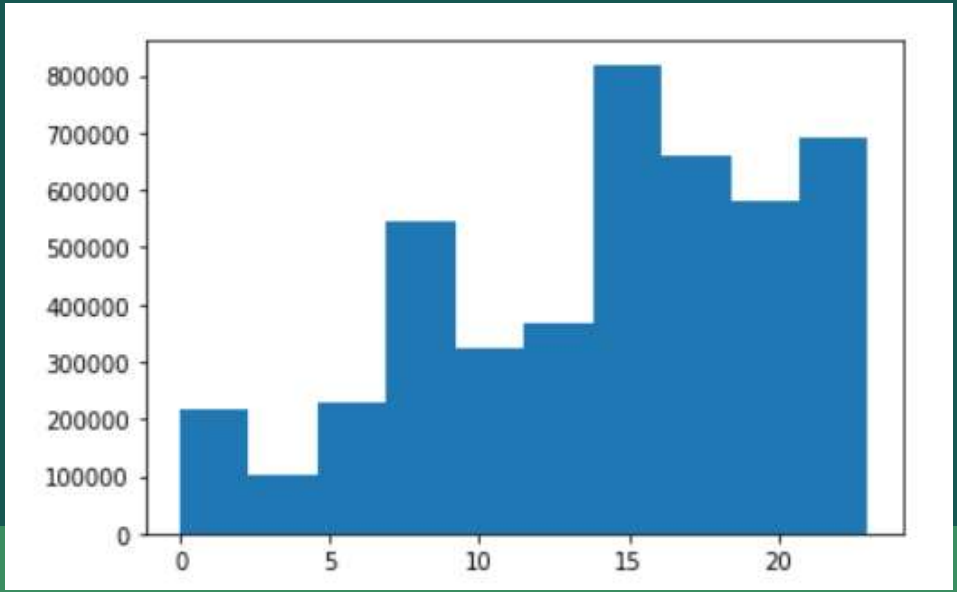


ANALYSIS OF JOURNEY BY WEEKDAYS

ANALYSING WHICH MONTH HAS MAXIMUM RIDES

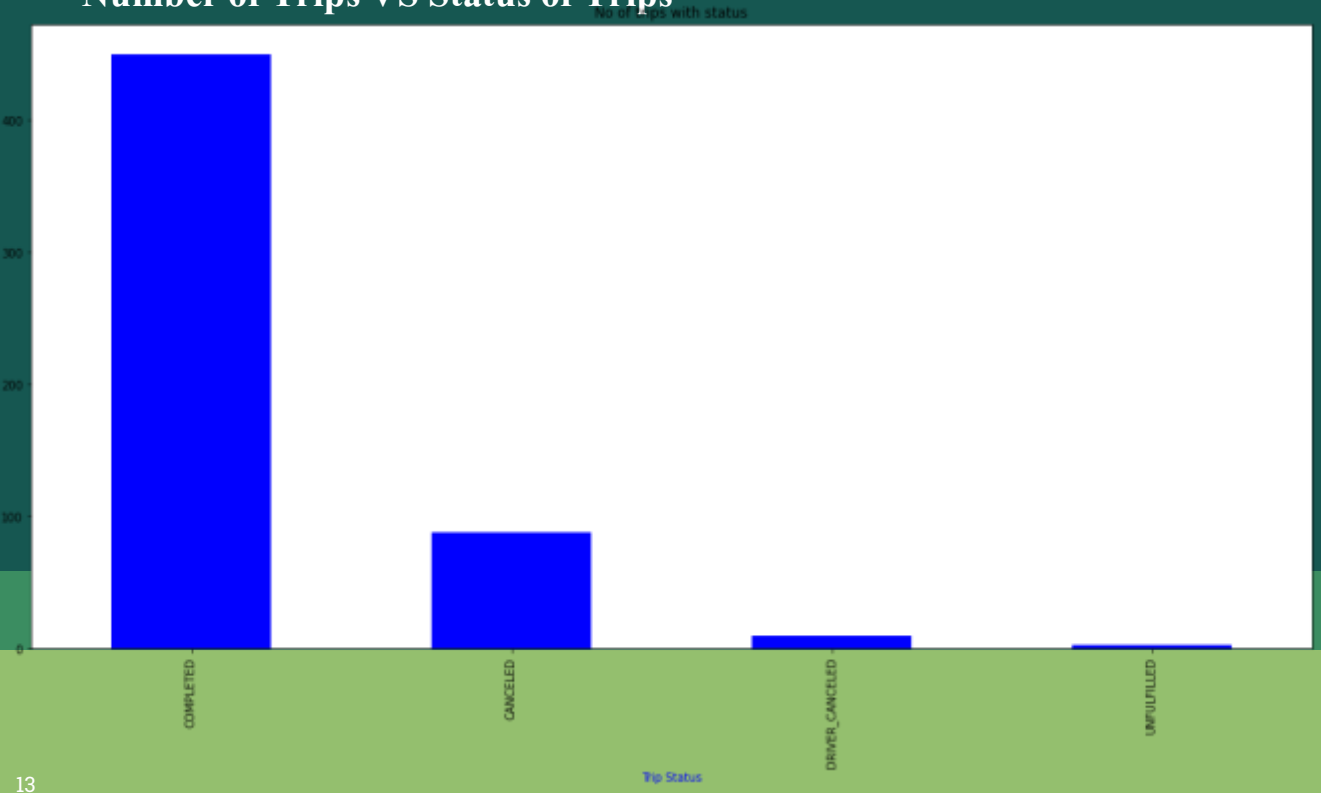


ANALYSIS OF JOURNEY BY HOURS



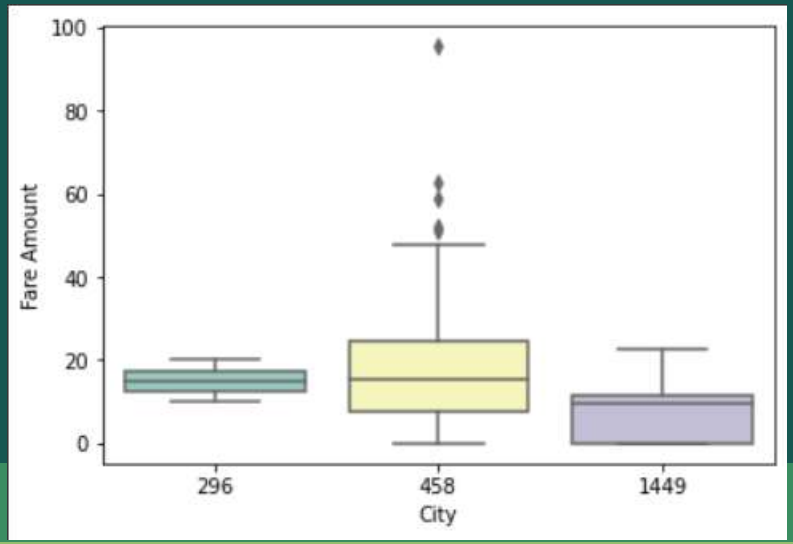
Fare of the drives

Number of Trips VS Status of Trips

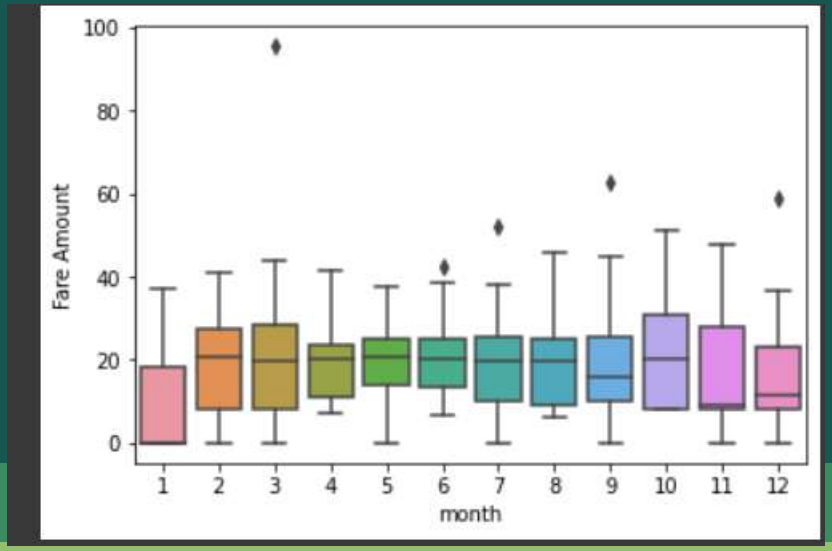


Box Plot for Fare Amount

The graph shows 3 unique cities and how their Fare Amount Vary



The graph shows Month vs Fare Amount



Outlier Analysis

Inter Quartile Range

```
[ ]  # IQR
     Q1 = np.percentile(df['Fare Amount'], 25,
          interpolation = 'midpoint')

     Q3 = np.percentile(df['Fare Amount'], 75,
          interpolation = 'midpoint')
     IQR = Q3 - Q1
     IQR

     16.735
```

```
[ ]  upper = Q3 +1.5*IQR

     lower = Q1 - 1.5*IQR
```

```
[ ]  # Above Upper bound
     upper = df['Fare Amount'] >= (Q3+1.5*IQR)

     print("Upper bound:",upper)
     print(np.where(upper))

     # Below Lower bound
```

Z score

```
[ ]  z = np.abs(stats.zscore(df['Fare Amount']))
     print(z)
```

```
[ ]  threshold = 3

     # Position of the outlier
     outlier = np.where(z > threshold)
     print(outlier)

     (array([  2, 181, 210]),)
```

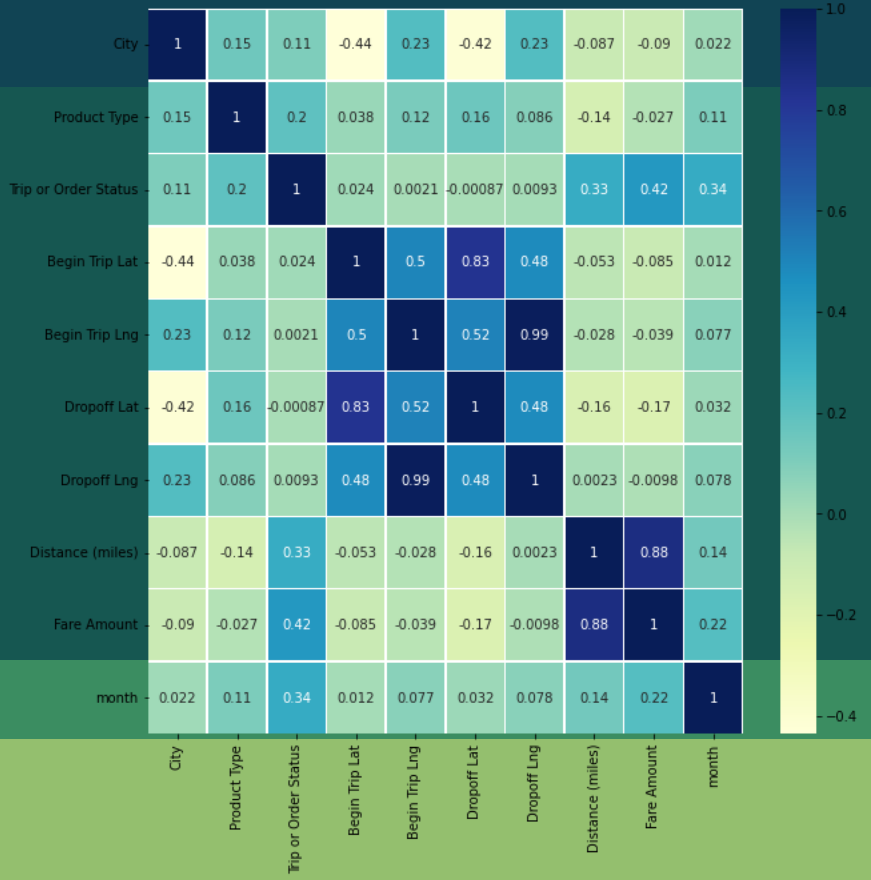
```
[ ]  df.shape

     (551, 16)
```

Heatmap for different attributes and Fare Amount

The features that have more influence on Fare Amount are:

1. Trip or Order Status
2. Distance(miles)
3. Month



MODEL BUILDING

Regression Model

```
[ ] df_new.head()
```

	Distance (miles)	Trip or Order Status	month
1	5.31	1	1
2	5.90	1	1
4	2.54	1	12
5	6.17	1	12
7	6.71	1	12

```
[ ] y = df['Fare Amount']

[ ] from sklearn.linear_model import LinearRegression
    model = LinearRegression()

[ ] history = model.fit(df_new,y)

[ ] history.score(df_new,y)
    0.7908456577190438
```

ANN Model Building

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
y = y.values.reshape(-1,1)
```

```
y_scaled = scaler.fit_transform(y)
```

```
X_scaled.shape
```

```
(546, 3)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled, test_size = 0.25)
```

```
X_train.shape
```

```
(409, 3)
```

```
y_test.shape
```

```
(137, 1)
```

```
import tensorflow.keras
from keras.models import Sequential
from keras.layers import Dense
from sklearn.preprocessing import MinMaxScaler

model = Sequential()
model.add(Dense(25, input_dim=3, activation='relu'))
model.add(Dense(25, activation='relu'))
model.add(Dense(1, activation='linear'))
model.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_28 (Dense)             (None, 25)                100

dense_29 (Dense)             (None, 25)                650

dense_30 (Dense)             (None, 1)                 26

=================================================================
Total params: 776
Trainable params: 776
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(optimizer='adam', loss='mean_squared_error')
```

Model Analysis