UBER DATA ANALYSIS

Foundations of Data Analytics (CSE3505)
Fall Semester 2022-2023

Uber Data Analysis

Project Report submitted in partial fulfillment of the requirements for the course of

Foundations of Data Analytics (CSE3505)

by

Sajal Verma (20BCE1041)
Sathiyanarayan G S(20BCE1045)
Santhosh Ram G K(20BCE1091)

VIT UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ■ CHENNAI
www.vit.ac.in

Fall Semester 2022-2023

DECLARATION

We hereby declare that the project entitled **“Uber Data Analysis”** submitted by us in partial fulfillment of the requirements for the course “Foundations of Data Analytics-CSE3505”, is a record of bonafide work carried out by us under the course faculty of **Dr.R.Priyadarshini.** We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for any other course of this institute or of any other institute or university.

VIT®
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ▪ CHENNAI
www.vit.ac.in

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. PRIYADARSHINI R,** School of Computer Science and Engineering, for her consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Ganesan R,** Dean of School of Computer Science and Engineering, VIT Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our Head of the Department **Dr. Rekha D** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

ABSTRACT:

Data analytics has helped companies optimize and grow their performance for decades. Data analytics and visualization has aided us with several benefits, few of them being Identifying emerging trends, studying relationships and patterns in data, analysis in depth and cherry on top are the insights we draw from these patterns. It is a requirement of time that we study these concepts thoroughly for all the benefits it provides.

Uber has been a major source of travel for people living in urban areas. Some people don't have their vehicles while some don't drive their vehicles intentionally because of their busy schedule. With over 118 million users, 5 million drivers, and 6.3 billion trips with 17.4 million trips completed per day , Uber is the company behind the data for moving people and making deliveries hassle-free. How are drivers assigned to riders cost-efficiently, and how is dynamic pricing leveraged to balance supply and demand?

In this project, we will take you through Exploratory Data Analysis of Uber trips analysis using Python. We will perform data analysis on two datasets from Uber. The first dataset contains information about the rides taken by one particular user and the second contains Uber and Lyft mass dataset and the rides taken by Uber users will be used for price prediction purposes.

Keywords:

Uber, Exploratory Data Analysis, Price Prediction, Linear Regression.

TABLE OF CONTENTS

SERIAL NO	TITLE	PAGE NO.
1	INTRODUCTION	
2	LITERATURE SURVEY	
3	SYSTEM ARCHITECTURE AND DESIGN	
4	EXPLORATORY DATA ANALYSIS	
5	MODEL IMPLEMENTATION	
6	PERFORMANCE ANALYSIS	
7	CONCLUSION AND FUTURE WORK	
8	REFERENCES	

INTRODUCTION

We have used two datasets for analyzing Uber data.The first dataset contains information about 1155 rides of a single Uber user in 2016. The features include the trip date, source, destination, distance traveled, and purpose of the trip. This dataset is a good starting point for performing basic EDA. Based on this, we might also be able to generate some insights by relating the data to real-world events and user habits. Our goal is to explore this dataset and provide valuable insights.

The second expansive Uber and Lyft Dataset contains cab ride information and several other details about the trip environment for all the Uber, and Lyft rides taken in Boston, MA. We combined that data with weather data for better analysis. We use data from only the Uber rides here. Our goal is to predict the price using linear regression for this dataset.

For this project, we have used the Python language for coding and its libraries NumPy, Pandas for Data Manipulation and Matplotlib, Seaborn for Data Visualization.

LITERATURE REVIEW

Past few years have seen tremendous growth in uber related data analysis using machine learning. People are coming up with various methods to analyze uber related data such as A state in which the results, k-means clustering is used to estimate the most likely collection points at a given time and to predict the best hotspots of nightlife learning trends from previous Uber pickups. The center of the taxi service decides on the space of the area to be targeted for the pickup of passengers.

This can be justified by explaining that how the core of Uber and how it has impacted on tremendous growth.

• Bridging the supply demand gap
• Reduction in ETA
• Route Optimization

[1] Travel Time Estimation Accuracy in Developing Regions: An Empirical Case Study with Uber Data in Delhi-NCR

This paper investigates the quality of travel time estimates in the Indian capital city of Delhi and the National Capital Region (NCR). Using Uber mobile and web applications, we collect data about 610 trips from 34 Uber users. We empirically show the unpredictability of travel time estimates for Uber cabs. We also discuss the adverse effects of such unpredictability on passengers waiting for the cabs, leading to a whopping 28.4% of the requested trips being canceled. Our empirical observations differ significantly from the high accuracies reported in travel time estimation literature. These pessimistic results will hopefully trigger useful investigations in future on why the travel time estimates are mismatching the high accuracy levels reported in literature. This paper identifies and discusses the important problem of travel time estimation inaccuracies in developing countries.

[2] Predicting Short-Term Uber Demand in New York City Using Spatiotemporal Modeling:

Faghih, S.S recommends a recent modeling approach in Manhattan, New York City, to capture the demand for electronic mail services, particularly the Uber application. Uber collection data is added to the Manhattan TAD level and at 15-minute time intervals. This aggregation allows the implementation of a modern approach to spatio-temporal modeling to obtain a spatial and temporal understanding of the demand. During a typical day, two spacetime models were developed using Uber collection data, the STAR and STAR and MSPE turns determine the output of the models. The results of the MSPE have shown that it is recommended to use the Lasso-Star system instead of the star design. A comparison between the demand for yellow and uber taxis in 2014 and 2015 in New York shows that the demand for uber has increased.

[3] Rahul Pradhan, Praveen Kumar Mannepallli - Analyzing Uber Trips using PySpark:

Most of the organizations going on line the place the statistics generate will increase day by day. To develop commercial enterprise with this aggressive surrounding records evaluation is necessary. This analytics venture is very important to recognize the use of records analytics. Through initiatives like this, many organizations can recognize a number of complicated operations. Uber Data Analysis task permits us to recognize the complicated facts visualization of this large organization. It is developed with the assistance of the ‘R’ programming language. In this venture we analyze the Daily, Monthly and Yearly Uber Pickups in New York City. This mission is primarily based on Data Visualization that will inform you toward use of the ggplot2 library for perception of the data.

[4] Data Analysis of Uber and Lyft Cab Services:

Shashank H. understands how we can use machine learning in order to predict the cab fare from given source to destination before starting the cab ride. The model created is able to give us the predictions which are not exactly equal to the actual the price fluctuation is around the difference of ten to twenty rupees compared to the actual price. Since the model is good but not the best, we can improve the predictions of the model by using the Fine-tuning technique. If fine tuning is applied to the existing model, we are able to get higher accuracy than the proposed model.

[5] Investigating Uber price surges during a special event in Austin, TX

Junfeng Jiao proposed a study to evaluate the characteristics of Transportation Network Company (TNC) Uber's surge pricing during a special event. Using data collected using Uber's developer API over the 2015 Fourth of July weekend, this research investigated the form of price surge multipliers during periods of high demand. Regression models showed surge price was not correlated with ride wait time for July 3, July 4, or July 5, but it was correlated with ride request time in all three nights. July 4 had the strongest correlation and more instances of surge pricing, and those instances were greater in magnitude that the other evenings studied. This research has practical implications for transportation planners in that it reveals the obscurity of the price surge mechanisms. The unpredictability and lack of transparency surrounding surge pricing poses challenges for those working to incorporate TNCs into a city's transportation operations.

SYSTEM ARCHITECTURE AND DESIGN

There are 4 phases of System Architecture that are followed for this project.

1. Data collection
2. Data cleaning and processing
3. Model training
4. Testing and evaluation

We collected the dataset from Kaggle. Then we started cleaning our dataset, where we removed the duplicate values and NaN data. After that, we processed the data to make it suitable for insights generation. Data was visualized for providing insights. In linear regression, we are attempting to build a model that allows us to predict the value of new data, given the training data used to train our model.

EXPLORATORY DATA ANALYSIS

Data Exploration:

First and last 5 rows of the dataset:

```
# First 5 records
uber_df.head()
```

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit

```
# Last 5 records
uber_df.tail()
```

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site
1155	Totals	NaN	NaN	NaN	NaN	12204.7	NaN

Dimension and Size of the dataset:

```
# The  shape and size of data
print(uber_df.shape)
print (uber_df.size)
```

```
(1156, 7)
8092
```

The dataset contains 1156 rows and the size is 8092

```
# Data  type of the columns

uber_df.dtypes
```

```
START_DATE*     object
END_DATE*       object
CATEGORY*       object
START*          object
STOP*           object
MILES*         float64
PURPOSE*        object
```

The dataset contains 6 categorical data and 1 numerical data.

```
#Get the number of missing values in each column
uber_df.isnull().sum()
```

```
START_DATE*      0
END_DATE*        1
CATEGORY*        1
START*           1
STOP*            1
MILES*           0
PURPOSE*       503
dtype: int64
```

Purpose column contains 503 null values.

Data Cleaning:

```
uber_df[uber_df.duplicated()]
```

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
492	6/28/2016 23:34	6/28/2016 23:59	Business	Durham	Cary	9.9	Meeting

```
# Dropping the duplicates values

uber_df.drop_duplicates(inplace=True)
```

```
# Get the initial data with dropping the NA values
uber_df = uber_df.dropna()

#Get the shape of the dataframe after removing the null values
uber_df.shape
```

```
(652, 7)
```

43% of the dataset contained null values in purpose columns and 1 row containing duplicate values were dropped for analysis.

Exploring Start and Stop points:

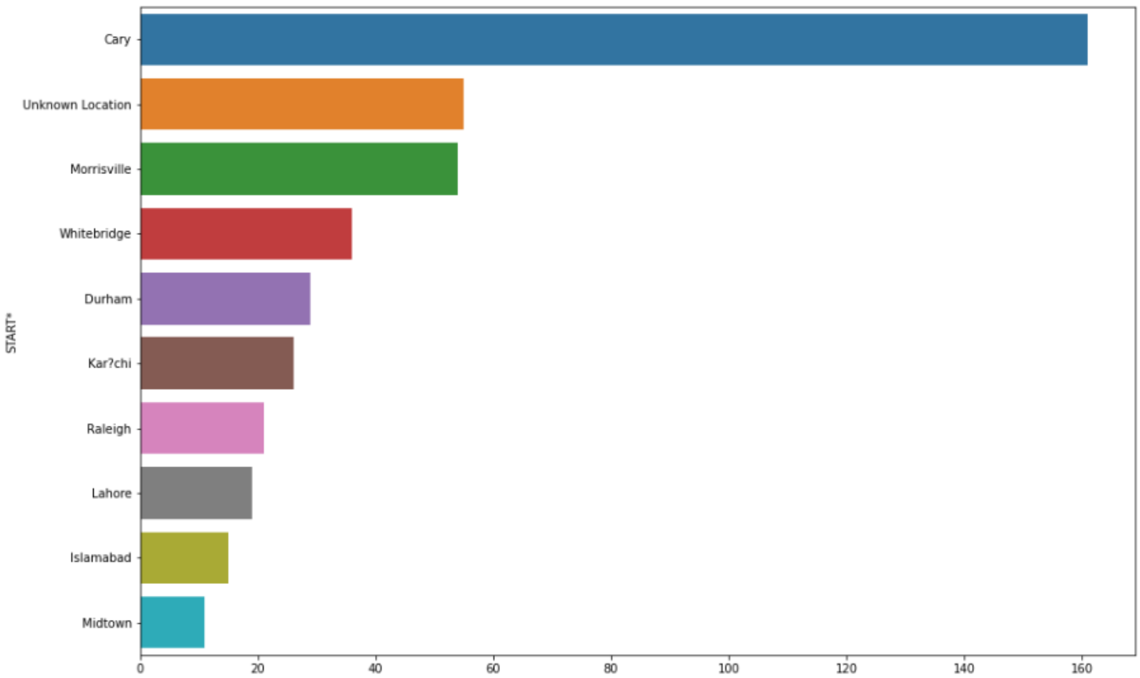
The top 10 starting and destination points were analyzed. A place named Cary was the most popular Starting and Destination point.

Top 10 Start points:

```
#Identify popular start destinations - top 10
uber_df['START*'].value_counts().head(10)
```

```
Cary                161
Unknown Location     55
Morrisville          54
Whitebridge          36
Durham               29
Kar?chi              26
Raleigh              21
Lahore               19
Islamabad            15
Apex                 11
```

```
plt.figure(figsize=(15,10))
sns.countplot(y="START*",order= pd.value_counts(uber_df['START*']).iloc[:10].index, data=uber_df)
plt.show()
```

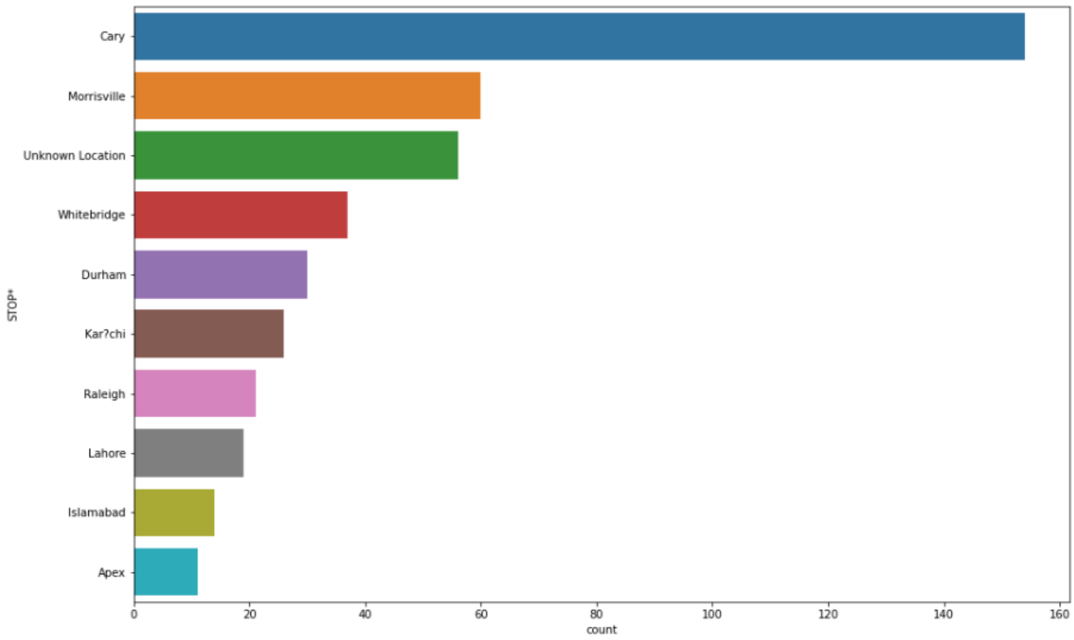


<u>Top 10 stop points:</u>

```
#Identify popular stop destinations - top 10
uber_df['STOP*'].value_counts().head(10)
```

```
Cary                154
Morrisville          60
Unknown Location     56
Whitebridge          37
Durham               30
Kar?chi              26
Raleigh              21
Lahore               19
Islamabad            14
Apex                 11
```

```
plt.figure(figsize=(15,10))
sns.countplot(y="STOP*",order= pd.value_counts(uber_df['STOP*']).iloc[:10].index, data=uber_df)
plt.show()
```



Distance between points:

Cary and Durham were the farthest points with 312.3 miles distance for the user and other top farthest distanced points are viewed.

```
#Find out most farthest start and stop pair -top10
#Dropping Unknown Location Value
uber_df2 = uber_df[uber_df['START*']!= 'Unknown Location']
uber_df2 = uber_df2[uber_df2['STOP*']!= 'Unknown Location']

uber_df2.groupby(['START*','STOP*'])['MILES*'].sum().sort_values(ascending=False).head(10)
```

```
START*        STOP*
Cary          Durham         312.3
Latta         Jacksonville   310.3
Cary          Morrisville    293.7
Durham        Cary           288.5
Raleigh       Cary           269.5
Morrisville   Cary           250.6
Cary          Cary           233.9
              Raleigh        230.4
Jacksonville  Kissimmee      201.0
Boone         Cary           180.2
Name: MILES*, dtype: float64
```

Cary and Durham are the farthest from each other

The user has traveled from Cary to Morrisville 52 times which is maximum. The other most used drive points are displayed below.

```
#Find out most popular start and stop pair - top10
uber_df2.groupby(['START*','STOP*']).size().sort_values(ascending=False).head(10)
```

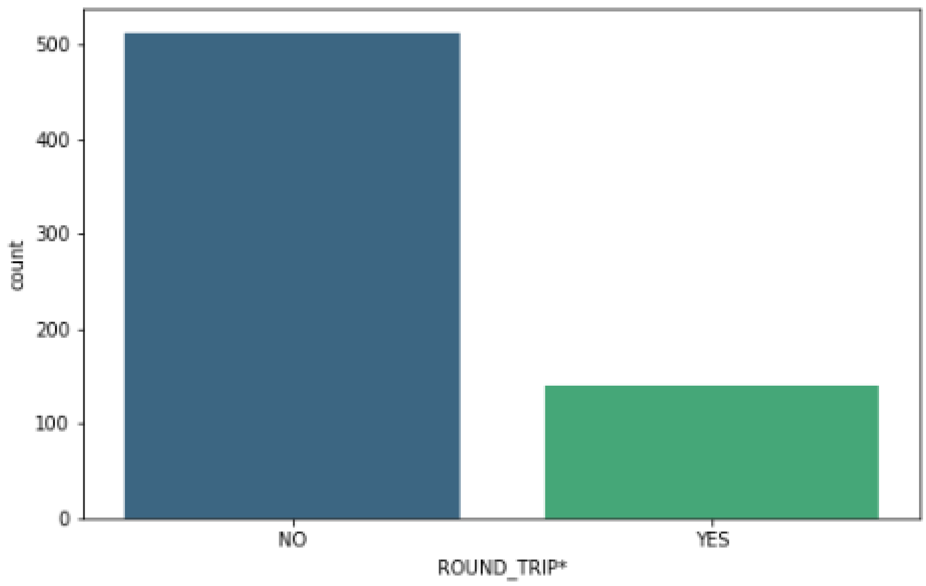
```
START*       STOP*
Cary         Morrisville    52
Morrisville  Cary           51
Cary         Cary           44
             Durham         30
Durham       Cary           28
Kar?chi      Kar?chi        20
Cary         Raleigh        17
Lahore       Lahore         16
Raleigh      Cary           15
Cary         Apex           11
dtype: int64
```

Number of Round Trips for the user is lesser.

```
# For this purpose, we need to make a function
plt.figure(figsize=(8,5))
def round(x):
    if x['START*'] == x['STOP*']:
        return 'YES'
    else:
        return 'NO'

uber_df['ROUND_TRIP*'] = uber_df.apply(round, axis=1)

sns.countplot(uber_df['ROUND_TRIP*'], order=uber_df['ROUND_TRIP*'].value_counts().index, palette='viridis')
plt.show()
```



Exploring date and time object:

For exploring the date and time, it is first converted into Month/Day/Year Hours:Minutes format using strptime function. Then the difference of time between end point and start point gives the duration of the ride.

```
# Convert the START DATE and END_DATE in string format to datetime object

uber_df.loc[:, 'START_DATE*'] = uber_df['START_DATE*'].apply(lambda x: pd.datetime.strptime(x, '%m/%d/%Y %H:%M'))
uber_df.loc[:, 'END_DATE*'] = uber_df['END_DATE*'].apply(lambda x: pd.datetime.strptime(x, '%m/%d/%Y %H:%M'))
```

```
#Calculate the duration for the rides
uber_df['DIFF'] = uber_df['END_DATE*'] - uber_df['START_DATE*']
```

The average trip time is 23 minutes and ride durations range from 2 minutes to 330 minutes.

```
uber_df['DIFF'].describe()
```

```
count    652.000000
mean      23.395706
std       25.789348
min        2.000000
25%       11.000000
50%       17.500000
75%       28.000000
max      330.000000
Name: DIFF, dtype: float64
```

Getting Hour,Day,Month and Year are captured in separate columns and analysis are done with respect to those columns.

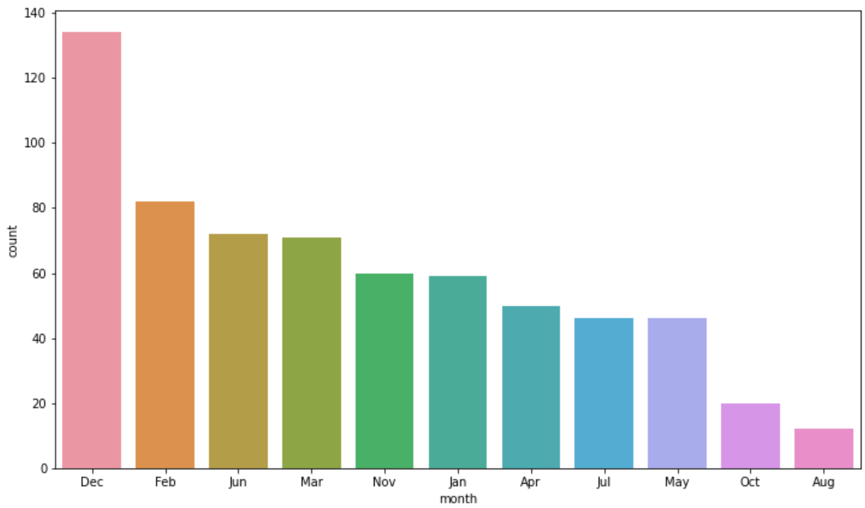
```
#Capture Hour, Day, Month and Year of Ride in a separate column
uber_df['month'] = pd.to_datetime(uber_df['START_DATE*']).dt.month
uber_df['Year'] = pd.to_datetime(uber_df['START_DATE*']).dt.year
uber_df['Day'] = pd.to_datetime(uber_df['START_DATE*']).dt.day
uber_df['Hour'] = pd.to_datetime(uber_df['START_DATE*']).dt.hour
```

Monthly wise Rides:

```
plt.figure(figsize=(12,7))
sns.countplot(uber_df['month'],order=pd.value_counts(uber_df['month']).index)
plt.show()
```

```
#Extract the total number of trips per month, weekday
print(uber_df['month'].value_counts())
print(uber_df['day_of_week'].value_counts())
```

```
Dec    134
Feb     82
Jun     72
Mar     71
Nov     60
Jan     59
Apr     50
Jul     46
May     46
Oct     20
Aug     12
Name: month, dtype: int64
Fri     125
Tue      93
Thur     92
Sun      87
Mon      87
Wed      85
Sat      83
Name: day_of_week, dtype: int64
```



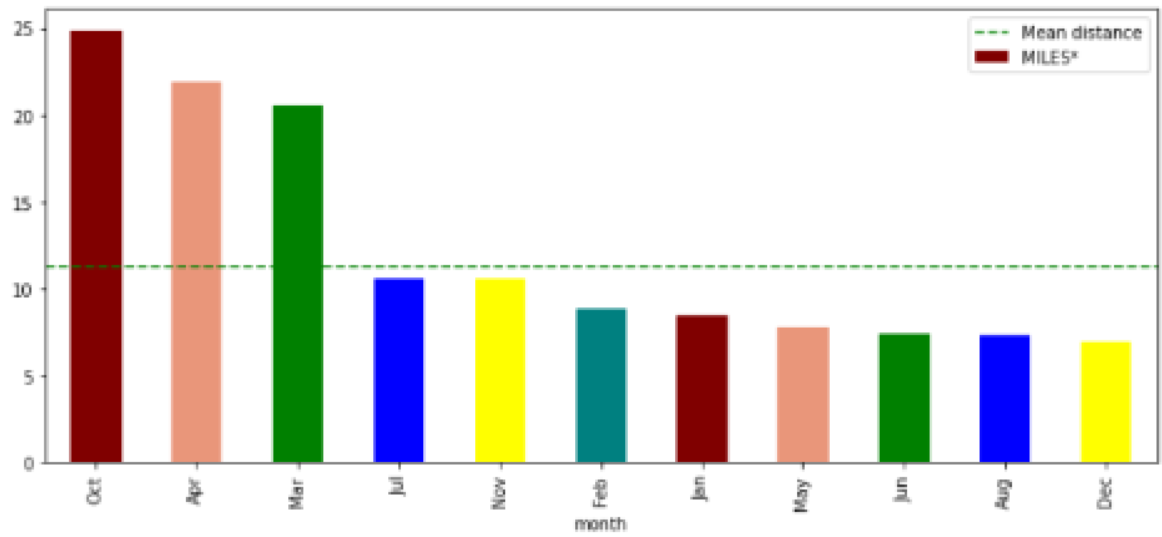
December has the highest number of rides and August has the least rides.

```
#Getting the average distance covered per month
uber_df.groupby('month').mean()['MILES*'].sort_values(ascending = False)
```

```
month
Oct    24.840000
Apr    21.898000
Mar    20.505634
Jul    10.615217
Nov    10.590000
Feb     8.868293
Jan     8.486441
May     7.793478
Jun     7.376389
Aug     7.341667
Dec     6.898507
Name: MILES*, dtype: float64
```

October has the highest average distance covered of 24.8miles and the least is December. From this we can infer that user has done many short distance rides in December.

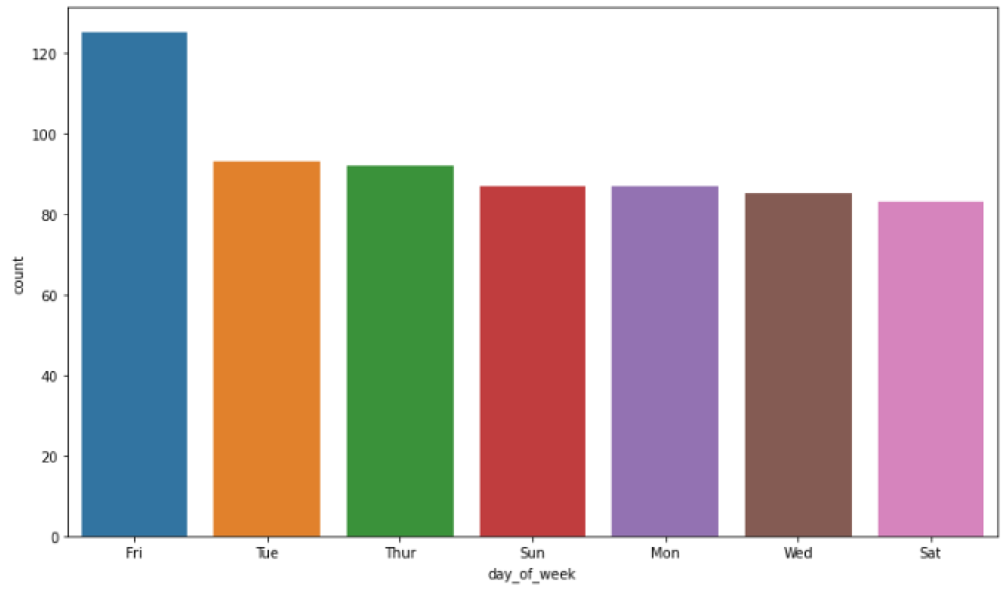
```
plt.figure(figsize=(12,5))
uber_df.groupby('month').mean()['MILES*'].sort_values(ascending = False).plot.bar(color=['maroon','darksalmon','green','blue','yello
w','teal'])
plt.axhline(uber_df['MILES*'].mean(), linestyle='--', color='green', label='Mean distance')
plt.legend()
plt.show()
```



Day wise Rides:

Friday has the most rides and all other days have a similar number of rides.

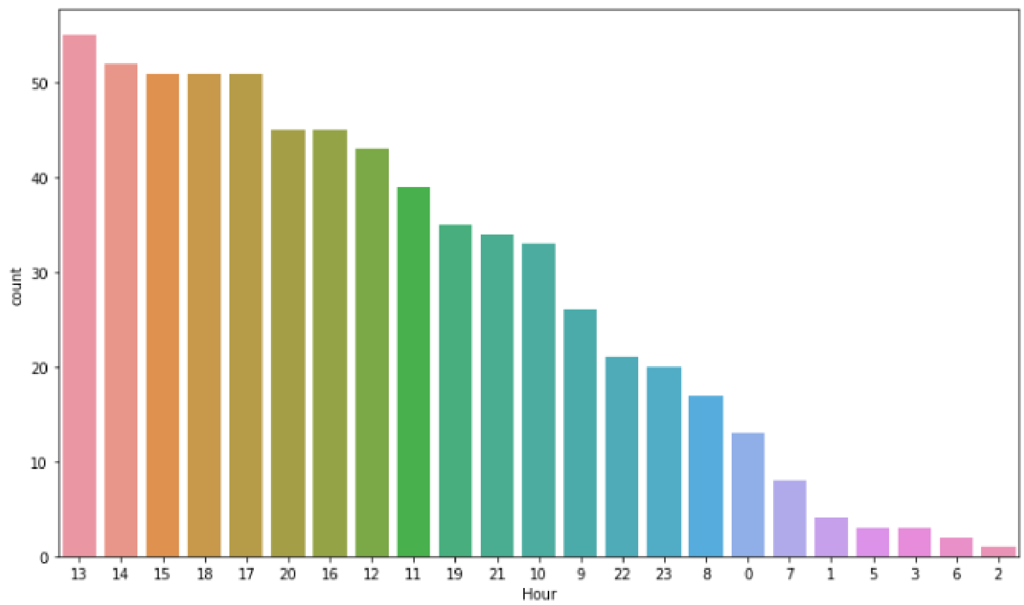
```
plt.figure(figsize=(12,7))
sns.countplot(uber_df['day_of_week'],order=pd.value_counts(uber_df['day_of_week']).index)
plt.show()
```



Hour wise Rides:

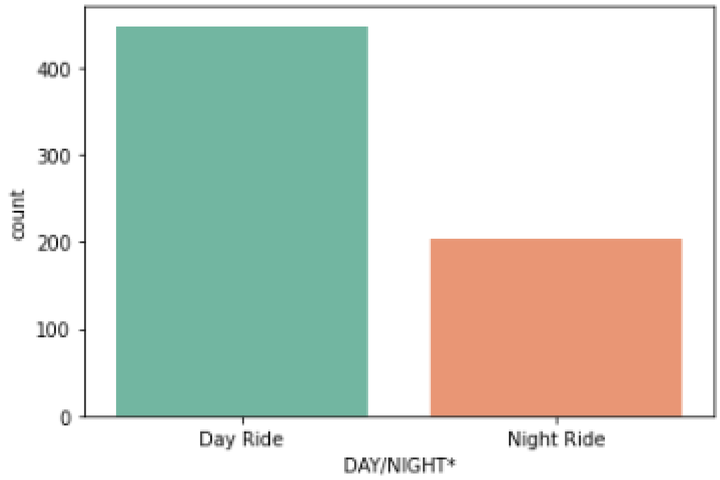
The user has undergone most of the rides in evening and lesser rides in early morning.

```
plt.figure(figsize=(12,7))
sns.countplot(uber_df['Hour'],order=pd.value_counts(uber_df['Hour']).index)
plt.show()
```

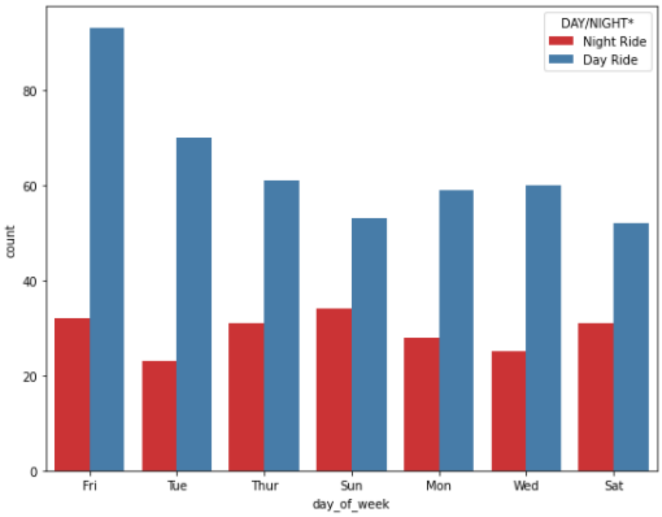


We splitted the rides to Day rides and night rides where Day rides are done before 6 P.M and Night rides are after 6PM. We found thay day rides were higher compared to night rides for this user.

```python
# Day Time or Night time
a = pd.to_datetime(['18:00:00']).time
uber_df['DAY/NIGHT*'] = uber_df.apply(lambda x : 'Night Ride' if x['START_DATE*'].time() > a else 'Day Ride', axis=1)
sns.countplot(uber_df['DAY/NIGHT*'], palette='Set2' , order = uber_df['DAY/NIGHT*'].value_counts().index)
plt.show()
```




```python
plt.figure(figsize=(9,7))
sns.countplot(uber_df['day_of_week'], hue=uber_df['DAY/NIGHT*'], palette='Set1' ,
              order=uber_df['day_of_week'].value_counts().index)
plt.show()
```



MODEL IMPLEMENTATION

The project has implemented the linear regression model for prediction of price.

Linear regression:

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on the kind of relationship between dependent and independent variables they are considering, and the number of independent variables getting used.

We have used two datasets: car data and weather data which is concat as a single data and linear regression is performed

```
cab_data
```

	distance	cab_type	time_stamp	destination	source	price	surge_multiplier	id	product_id	name
0	0.44	Lyft	1544952607890	North Station	Haymarket Square	5.0	1.0	424553bb-7174-41ea-aeb4-fe06d4f4b9d7	lyft_line	Shared
1	0.44	Lyft	1543284023677	North Station	Haymarket Square	11.0	1.0	4bd23055-6827-41c6-b23b-3c491f24e74d	lyft_premier	Lux
2	0.44	Lyft	1543366822198	North Station	Haymarket Square	7.0	1.0	981a3613-77af-4620-a42a-0c0866077d1e	lyft	Lyft
3	0.44	Lyft	1543553582749	North Station	Haymarket Square	26.0	1.0	c2d88af2-d278-4bfd-a8d0-29ca77cc5512	lyft_luxsuv	Lux Black XL
4	0.44	Lyft	1543463360223	North Station	Haymarket Square	9.0	1.0	e0126e1f-8ca9-4f2e-82b3-50505a09db9a	lyft_plus	Lyft XL
...	...	...	...	...	...	...	...	...	...	...
693066	1.00	Uber	1543708385534	North End	West End	13.0	1.0	616d3611-1820-450a-9845-a9ff304a4842	6f72dfc5-27f1-42e8-84db-ccc7a75f6969	UberXL
693067	1.00	Uber	1543708385534	North End	West End	9.5	1.0	633a3fc3-1f86-4b9e-9d48-2b7132112341	55c66225-fbe7-4fd5-9072-eab1ece5e23e	UberX
693068	1.00	Uber	1543708385534	North End	West End	NaN	1.0	64d451d0-639f-47a4-9b7c-6fd92fbd264f	8cf7e821-f0d3-49c6-8eba-e679c0ebcf6a	Taxi
693069	1.00	Uber	1543708385534	North End	West End	27.0	1.0	727e5f07-a96b-4ad1-a2c7-9abc3ad55b4e	6d318bcc-22a3-4af6-bddd-b409bfce1546	Black SUV
693070	1.00	Uber	1543708385534	North End	West End	10.0	1.0	e7fdc087-fe86-40a5-a3c3-3b2a8badcbda	997acbb5-e102-41e1-b155-9df7de0a73f2	UberPool

```
cab_data.columns
```

```
Index(['distance', 'cab_type', 'time_stamp', 'destination', 'source', 'price',
       'surge_multiplier', 'id', 'product_id', 'name', 'datetime'],
      dtype='object')
```

Testing and training sets:

```
x1=a[['distance', 'temp','clouds', 'pressure', 'humidity','wind','rain','day','hour','surge_multiplier','clouds']]
y1=a['price']
```

```
# Using Skicit-learn to split data into training and testing sets
from sklearn.model_selection import train_test_split
# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x1, y1, test_size = 0.25, random_state = 42)
```

```
linear=LinearRegression()
linear.fit(x_train,y_train)
linear.score(x_test, y_test)
```

```
0.1563858041768127
```

Linear coefficients of the columns.

```
linear.coef_
```

```
array([ 2.54489026e+00, -1.64620938e-14, -1.12950771e-13,  1.46767857e-14,
        6.35578344e-13,  2.76748123e-13, -6.67138380e-13,  1.54338424e+01,
        3.55271368e-15,  2.47433849e+01, -1.26502412e-13])
```

Actual VS Predicted values.

```
predictions=linear.predict(x_test)
print(predictions)
```

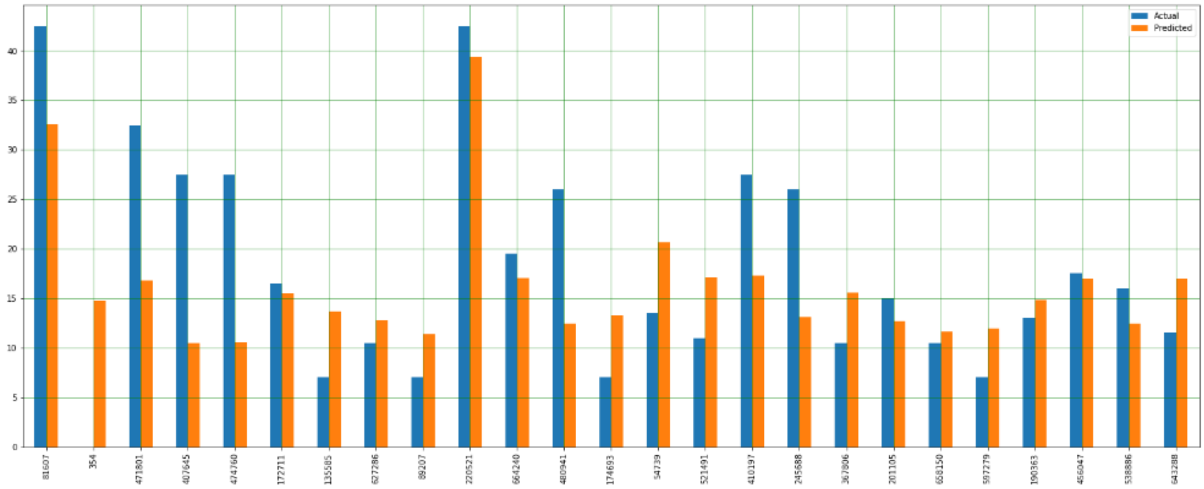
```
[32.5479163  14.7556079  16.81696887 ... 11.82898386 11.85443274
 13.22867338]
```

```
df = pd.DataFrame({'Actual': y_test, 'Predicted': predictions})
df
```

	Actual	Predicted
81607	42.5	32.547916
354	0.0	14.755608
471801	32.5	16.816969
407645	27.5	10.480192
474760	27.5	10.556539
...	...	...
538489	7.5	10.709232
579511	13.5	15.519075
5421	9.0	11.828984
279982	8.0	11.854433
236315	27.5	13.228673

174837 rows × 2 columns

```
df1 = df.head(25)
df1.plot(kind='bar',figsize=(26,10))
plt.grid(which='major', linestyle='-', linewidth='0.5', color='green')
plt.grid(which='minor', linestyle=':', linewidth='0.5', color='black')
plt.show()
```



PERFORMANCE ANALYSIS

RMSE:

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. Root mean square error is commonly used in climatology, forecasting, and regression analysis to verify experimental results. Predicting the price using the given columns was inaccurate since the RMSE is 9. From this we can get that Uber has confidential attributes in determining the price of the rides which is their success and one cannot find it accurately.

```
from sklearn.metrics import mean_absolute_error as mae
print(mae(y_test,predictions))
```

```
7.407742219419063
```

```
from sklearn.metrics import mean_squared_error as mse
mse(y_test,predictions)
np.sqrt(mse(y_test,predictions))
```

```
9.238360809376323
```

CONCLUSION AND FUTURE WORK

Through this project, we gained knowledge of various complex operations performed in data visualization. It enabled us to recognize the patterns in data of such a huge organization and provide critical insights of untapped information. Exploring this data gave confidence that we can generate insights from any dataset to derive conclusions. Uber data analysis is Exploratory Data Analysis.

We learned how to do Linear Regression in python and how to analyze the performance of it. We attempted to predict the price using linear regression but it was average which is the real success of Uber with the dynamic priced algorithm. Future works will be trying to find datasets containing specific columns or finding real world data to increase efficiency in predicting the ride prices.

REFERENCES

https://www.analyticsvidhya.com

https://w3techs.com/technologies/details/pl-python/

https://trends.builtwith.com/framework/Python

https://www.geeksforgeeks.org/ml-linear-regression/

Jasmine Lee. April 28. 2017. How Uber leverages Supply and demand in their pricing model.