K L UNIVERSITY

FRESHMAN ENGINEERING DEPARTMENT

A Project Based Lab Report

On

UBER DATA ANALYSIS AND PREDICT THE DEMAND

SUBMITTED BY:

ID NUMBER	NAME
2000030945	S.LAKSHMI PRASANNA
2000031007	T.SAI MRUDHULA
2000031013	Y.SAI BHARGHAV KUMAR

UNDER THE ESTEEMED GUIDANCE OF

Dr P Raja Rajeswari

ASSOCIATE PROFESSOR

KL UNIVERSITY

Green fields, Vaddeswaram – 522 502

Guntur Dt., AP, India.

DEPARTMENT OF BASIC ENGINEERING SCIENCES

CERTIFICATE

This is to certify that the project based laboratory report entitled CREDIT ANALYSIS LOAN PREDICTON submitted by Mr./Ms**.** **(PRASANNA,T.SAIMRUDHULA,SAIBHARGHAV)** bearing Regd. No(**2000030945,2000031007,2000031013**) to the **Department of Computer science Engineering, KL University** in partial fulfillment of the requirements for the completion of a project in "**ARTIFICIAL INTELLIGENCE AND DATA SCIENCE** "course in II B Tech II Semester, is a bonafide record of the work carried out by him/her under my supervision during the academic year 2021-2022.

PROJECT SUPERVISOR

Dr P Raja Rajeswari

HEAD OF THE DEPARTMENT

Pavan kumar

ACKNOWLEDGEMENTS

It is great pleasure for me to express my gratitude to our honorable President **Sri. Koneru Satyanarayana**, for giving the opportunity and platform with facilities in accomplishing the project based laboratory report.

I express the sincere gratitude to our director <name> for his administration towards our academic growth.

I express sincere gratitude to our Coordinator and HOD-BES **Dr.** for her leadership and constant motivation provided in successful completion of our academic semester. I record it as my privilege to deeply thank for providing us the efficient faculty and facilities to make our ideas into reality.

I express my sincere thanks to our project supervisor **RAJA RAJESWARI** for his/her novel association of ideas, encouragement, appreciation and intellectual zeal which motivated us to venture this project successfully.

Finally, it is pleased to acknowledge the indebtedness to all those who devoted themselves directly or indirectly to make this project report success.

Name : Prasanna, T.sai mrudhula, Sai bharghav

ID NO : 2000030945 , 2000031007, 2000031013

ABSTRACT

Uber company offer passenger boarding services that allow users to rent cars with drivers through websites or mobile apps. Whether traveling a short distance or traveling from one city to another, these services have helped people in many ways.

Data analytics has helped companies optimize and grow their performance for decades.In this project we use the dataset of uber for analysing the data

Uber data analysis helps the company to understand the back ground of various operations . With the help of visualization companies or people using the product can easily understand the benifits they understand the complex data and gain insights that would help to take decisions.

In this work, we build multiple machine learning models that increase the efficiency and sensitivity of uber analysis using descriptive and predictive analytics.

INDEX

INTRODUCTION

Uber is an international company located in 69 countries and around 900 cities around the world. Lyft, on the other hand, operates in approximately 644 cities in the US and 12 cities in Canada alone. However, in the US, it is the second-largest passenger company with a market share of 31%.

From booking a taxi to paying a bill, both services have similar features. But there are some exceptions when the two passenger services reach the neck. The same goes for prices, especially **Uber's** "**surge**" and "Prime Time" in Lyft. There are certain limitations that depend on where service providers are classified.

Many articles focus on algorithm/model learning, data purification, feature extraction, and fail to define the purpose of the model. Understanding the business model can help identify challenges that can be solved using analytics and scientific data. In this article, we go through the **Uber** Model, which provides a framework for end-to-end prediction analytics of **Uber** data prediction sources.

THEORICAL BACKGROUND

DATA ANALYSIS:

Data analysis is the process of collecting, modeling, and analyzing data to extract insights that support decision-making. There are several methods and techniques to perform analysis depending on the industry and the aim of the investigation.

Data Analysis is one aspect of Data Science that is all about analyzing data for different kinds of purposes.

Data Analysis tools: R, Python, Statistics, SAS, Jupyter, R Studio, MATLAB, Excel, RapidMiner.

TYPES OF DATA ANALYSIS:

There are 6 types

1. ***Descriptive Analysis***
2. ***Exploratory Analysis***
3. ***Inferential Analysis***
4. ***Predictive Analysis***
5. ***Causal Analysis***
6. ***Mechanistic Analysis***

- The Data Analytics Lifecycle is designed specifically for Big Data problems and data science projects.
- The lifecycle has six phases, and project work can occur in several phases at once.
- For most phases in the lifecycle, the movement can be either forward or backward.
- This iterative depiction of the lifecycle is intended to more closely portray a real project, in which aspects of the project move forward and may return to earlier stages as new information is uncovered and team members learn more about various stages of the project.
- This enables participants to move iteratively through the process and drive toward operationalizing the project work.

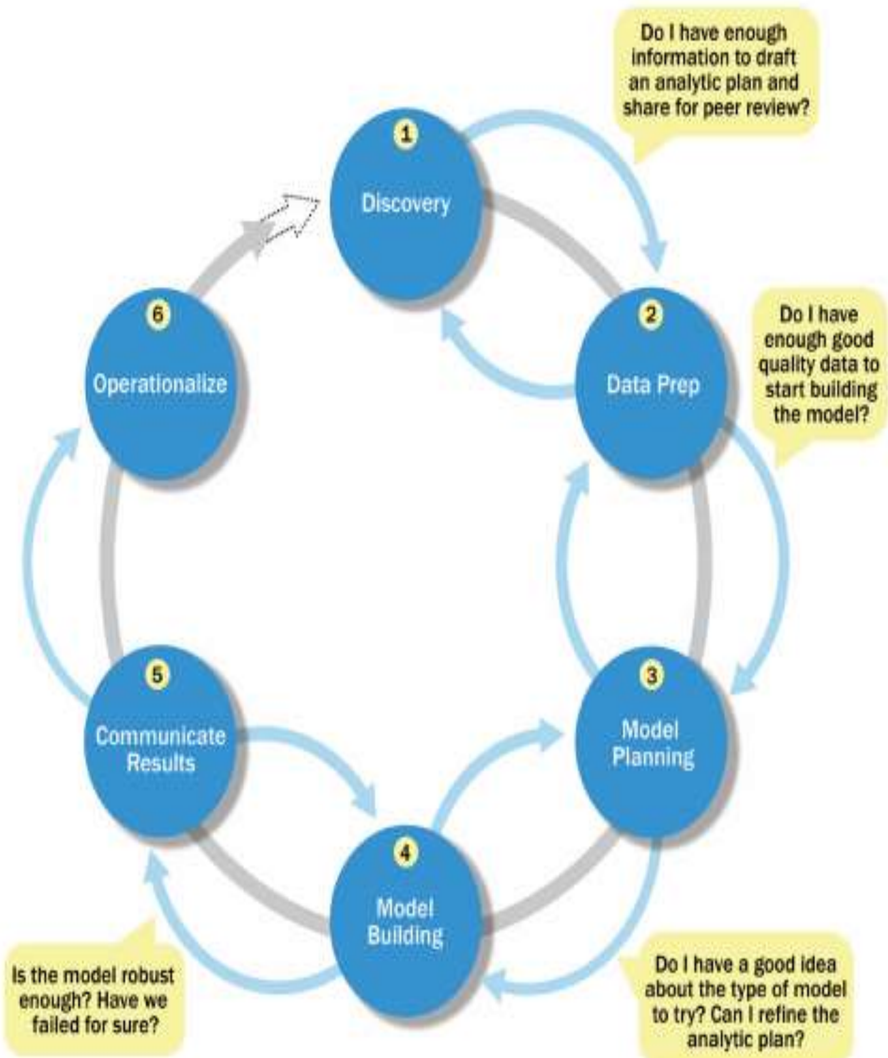
SYSTEM REQUIREMENTS

- A laptop or pc with windows i5 configuration

SOFTWARE REQUIREMENTS:

- **Python 3.9 version**
- **Jupyter notebook**
- **Google colabs**

FLOW CHART



1
Discovery
Do I have enough information to draft an analytic plan and share for peer review?
2
Data Prep
Do I have enough good quality data to start building the model?
3
Model Planning
Do I have a good idea about the type of model to try? Can I refine the analytic plan?
4
Model Building
Is the model robust enough? Have we failed for sure?
5
Communicate Results
6
Operationalize

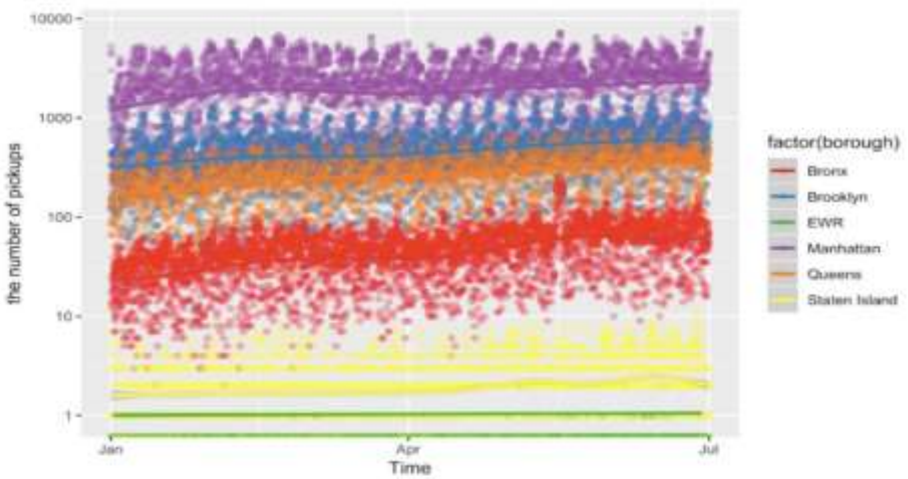
DATA ANALYTICS : EDA and PLOTTING

Before conducting exploratory data analysis, I removed missing value and changed the format of date .

In this section, I mainly studied the relationship between pickups with different features. I divided features into 3 categories, borough, time and weather information. User can access the second Tab "Visualization" to discover the relationship between feature. I will describe this section in brief.

Pickups with boroughs

In this part, I mainly studied the relationship between pickups in different boroughs for various time periods, 6 months, 1 day. User can either select 'Month' or 'Hour to check the distribution of pickup number. The following figure plots the pickups in 6 months. Different color represents different boroughs.



Pickups vs Borough

CODING

Uber Data Analysis

- Defining the problem statement
- Collecting the data
  - Kaggle
- Exploratory data analysis
- Feature engineering
- Modelling
- Testing

1.Defining the problem statement

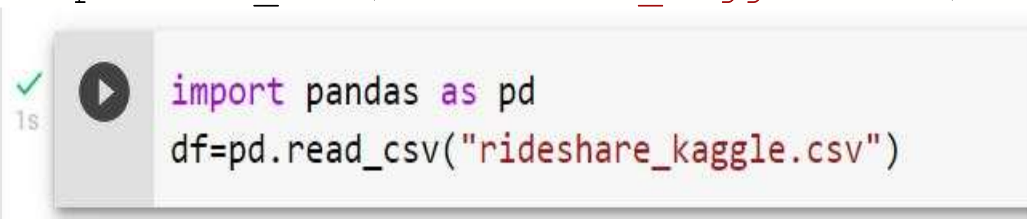
In this project, we study the data of Uber which is present in tabular format in which we use different libraries like numpy, pandas and matplotlib and different machine learning algorithms.

We study different columns of the table and try to co-relate them with others and find a relation between those two.

We try to find and analyze those key factors like date, month etc which helps Uber Company to enhance their business by focusing on those services and make required changes.

2. Collecting the data

```
import pandas as pd
df=pd.read_csv("rideshare_kaggle.csv")
```



File is taken from Kaggle dataset

Link to download the dataset:

https://www.kaggle.com/datasets/brllrb/uber-and-lyft-dataset-boston-ma?resource=download

3. Exploratory data analysis

- `df.head()`

[4] df.head()

	id	timestamp	hour	day	month	datetime	timezone	source	destination	cab_type	...	precipIntensityMax	uvIndexTime
	424553bb-7174-41ea-aeb4-fe06d4f4b9d7	1.544953e+09	9	16	12	2018-12-16 09:30:07	America/New_York	Haymarket Square	North Station	Lyft	...	0.1276	1544979600
	4bd23055-6827-41c6-b23b-3c491f24e74d	1.543284e+09	2	27	11	2018-11-27 02:00:23	America/New_York	Haymarket Square	North Station	Lyft	...	0.1300	1543251600
	981a3613-77af-4620-a42a-0c0866077d1e	1.543367e+09	1	28	11	2018-11-28 01:00:22	America/New_York	Haymarket Square	North Station	Lyft	...	0.1064	1543338000
	c2d88af2-d278-4bfd-a8d0-29ca77cc5512	1.543554e+09	4	30	11	2018-11-30 04:53:02	America/New_York	Haymarket Square	North Station	Lyft	...	0.0000	1543507200
	e0126e1f-8ca9-4f2e-[illegible]	1.543463e+09	3	29	11	2018-11-29	America/New_York	Haymarket [illegible]	North Station	Lyft	...	0.0001	1543420800

- df.shape

[6] df.shape

0s

(31629, 57)

- df.size

[27] df.size

4508757

- df.index

```
df.index
```

RangeIndex(start=0, stop=79101, step=1)

- df.columns

```
df.columns
```

```
Index(['id', 'timestamp', 'hour', 'day', 'month', 'datetime', 'timezone',
       'source', 'destination', 'cab_type', 'product_id', 'name', 'price',
       'distance', 'surge_multiplier', 'latitude', 'longitude', 'temperature',
       'apparentTemperature', 'short_summary', 'long_summary',
       'precipIntensity', 'precipProbability', 'humidity', 'windSpeed',
       'windGust', 'windGustTime', 'visibility', 'temperatureHigh',
       'temperatureHighTime', 'temperatureLow', 'temperatureLowTime',
       'apparentTemperatureHigh', 'apparentTemperatureHighTime',
       'apparentTemperatureLow', 'apparentTemperatureLowTime', 'icon',
       'dewPoint', 'pressure', 'windBearing', 'cloudCover', 'uvIndex',
       'visibility.1', 'ozone', 'sunriseTime', 'sunsetTime', 'moonPhase',
       'precipIntensityMax', 'uvIndexTime', 'temperatureMin',
       'temperatureMinTime', 'temperatureMax', 'temperatureMaxTime',
       'apparentTemperatureMin', 'apparentTemperatureMinTime',
       'apparentTemperatureMax', 'apparentTemperatureMaxTime'],
      dtype='object')
```

- df.info()

```
df.info()
```

```
 1   timestamp                    79101 non-null  float64
 2   hour                         79101 non-null  int64
 3   day                          79101 non-null  int64
 4   month                        79101 non-null  int64
 5   datetime                     79101 non-null  object
 6   timezone                     79101 non-null  object
 7   source                       79101 non-null  object
 8   destination                  79101 non-null  object
 9   cab_type                     79101 non-null  object
 10  product_id                   79101 non-null  object
 11  name                         79101 non-null  object
 12  price                        72879 non-null  float64
 13  distance                     79101 non-null  float64
 14  surge_multiplier             79101 non-null  float64
 15  latitude                     79101 non-null  float64
 16  longitude                    79101 non-null  float64
 17  temperature                  79101 non-null  float64
 18  apparentTemperature          79101 non-null  float64
 19  short_summary                79101 non-null  object
 20  long_summary                 79101 non-null  object
 21  precipIntensity              79101 non-null  float64
 22  precipProbability            79101 non-null  float64
 23  humidity                     79100 non-null  float64
 24  windSpeed                    79100 non-null  float64
 25  windGust                     79100 non-null  float64
```

- df.describe()

```
df.describe()
```

	timestamp	hour	day	month	price	distance	surge_multiplier	latitude	longitude	temperatur
count	7.910100e+04	79101.000000	79101.000000	79101.000000	72879.000000	79101.000000	79101.000000	79101.000000	79101.000000	79101.00000
mean	1.544032e+09	11.564810	18.007332	11.574203	16.572524	2.192278	1.015332	42.338381	-71.066173	39.55035
std	6.874164e+05	6.984642	9.981405	0.494466	9.364228	1.138368	0.096541	0.047610	0.020234	6.68921
min	1.543204e+09	0.000000	1.000000	11.000000	2.500000	0.020000	1.000000	42.214800	-71.105400	18.91000
25%	1.543441e+09	5.000000	13.000000	11.000000	9.000000	1.270000	1.000000	42.350300	-71.081000	36.50000
50%	1.543723e+09	12.000000	17.000000	12.000000	13.500000	2.160000	1.000000	42.351900	-71.063100	40.49000
75%	1.544817e+09	18.000000	28.000000	12.000000	22.500000	2.940000	1.000000	42.364700	-71.054200	43.57000
max	1.545161e+09	23.000000	30.000000	12.000000	92.000000	7.500000	3.000000	42.366100	-71.033000	57.22000

8 rows × 46 columns

- df.isnull().sum()

0s

```
df.isnull().sum()
```

```
id                        0
timestamp                 0
hour                      0
day                       0
month                     0
datetime                  0
timezone                  0
source                    0
destination               0
cab_type                  0
product_id                0
name                      0
price                  6222
distance                  0
surge_multiplier          0
latitude                  0
longitude                 0
temperature               0
apparentTemperature       0
short_summary             0
long_summary              0
precipIntensity           0
precipProbability         0
humidity                  1
```

4. Feature Engineering

Main goals of feature engineering

1)Preparing the proper input dataset, compatible with the machine learning algorithm requirements.

2) Improving the performance of machine learning models.

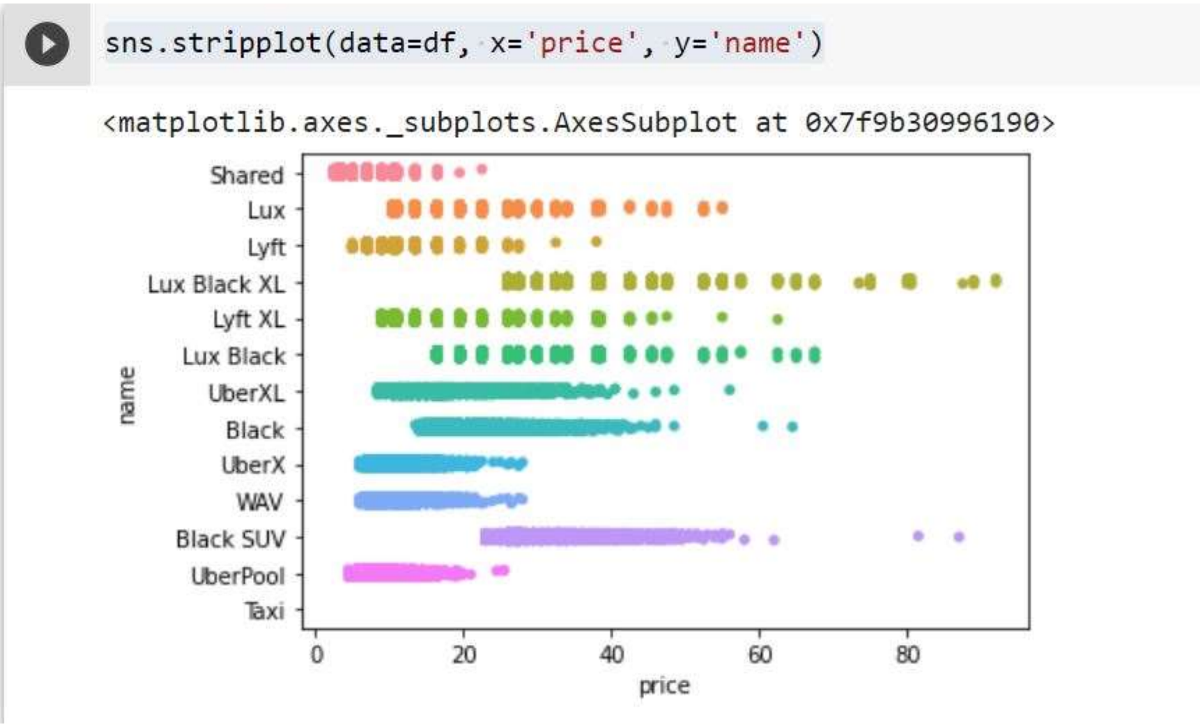
Plotting:

Importing libraries

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
import seaborn as sns
import pandas as pd
```

Strip plot:

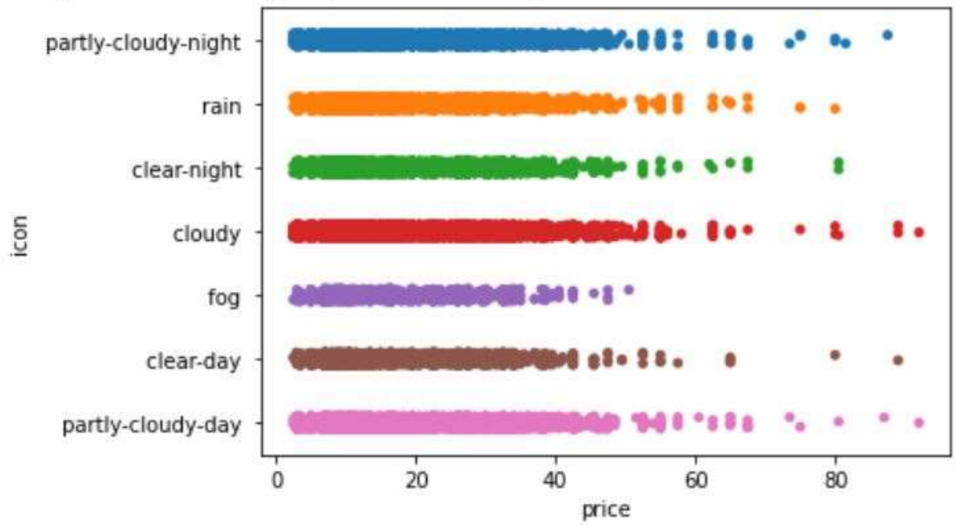
```
sns.stripplot(data=df, x='price', y='name')
```



```
sns.stripplot(data=df, x='price', y='icon')
```

```
sns.stripplot(data=df, x='price', y='icon')
```

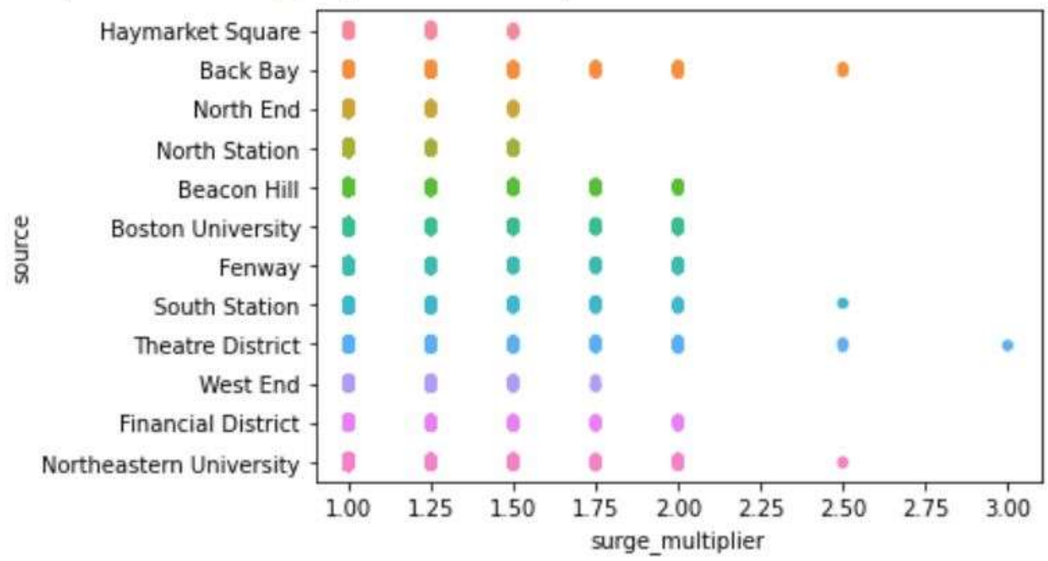
<matplotlib.axes._subplots.AxesSubplot at 0x7f9b30980150>




```
sns.stripplot(data=df, x='surge_multiplier', y=
'source')
```

```
sns.stripplot(data=df, x='surge_multiplier', y='source')
```

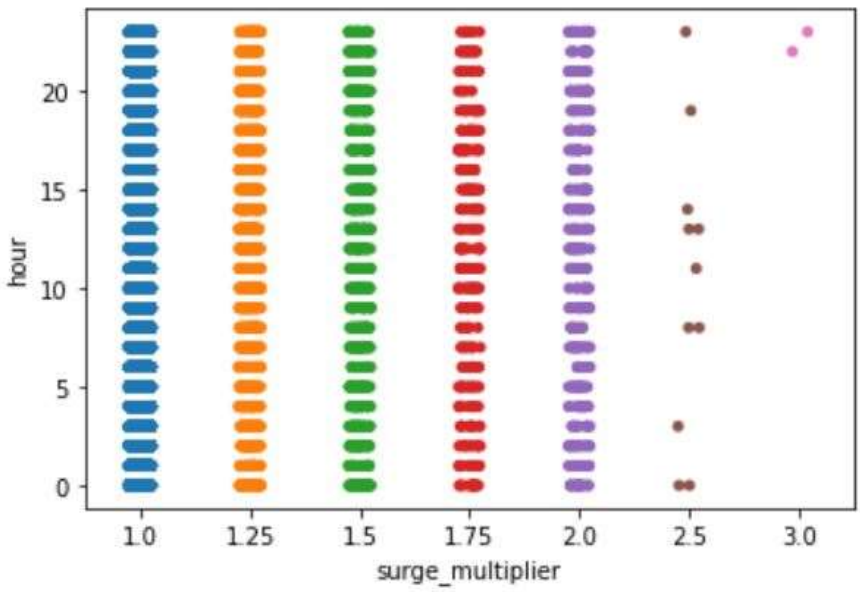
<matplotlib.axes._subplots.AxesSubplot at 0x7f9b303dbf50>



```
sns.stripplot(data=df, x='surge_multiplier', y=
'hour')
```

```
[16] sns.stripplot(data=df, x='surge_multiplier', y='hour')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9b303dca10>
```



Converting Timestamp to Datetime value

```
df['timestamp'].head()
```

```
df['timestamp'].head()
```

```
0     1.544953e+09
1     1.543284e+09
2     1.543367e+09
3     1.543554e+09
4     1.543463e+09
Name: timestamp, dtype: float64
```

```
from datetime import datetime
timestamp1 = 1544952608
timestamp2 = 1543284024
timestamp3 = 1543818483
timestamp4 = 1543594384
timestamp5 = 1544728504
```

```
dt_object1 = datetime.fromtimestamp(timestamp1)
dt_object2 = datetime.fromtimestamp(timestamp2)
dt_object3 = datetime.fromtimestamp(timestamp3)
dt_object4 = datetime.fromtimestamp(timestamp4)
dt_object5 = datetime.fromtimestamp(timestamp5)

print("dt_object =", dt_object1)
print("dt_object =", dt_object2)
print("dt_object =", dt_object3)
print("dt_object =", dt_object4)
print("dt_object =", dt_object5)
```

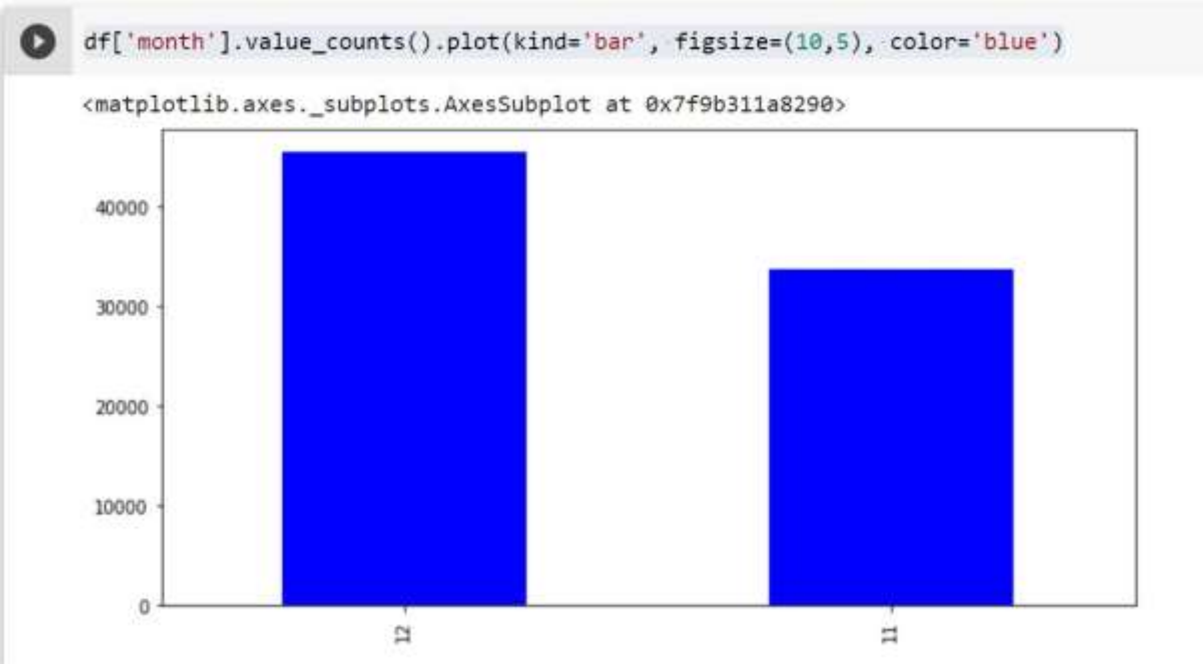
```
from datetime import datetime
timestamp1 = 1544952608
timestamp2 = 1543284024
timestamp3 = 1543818483
timestamp4 = 1543594384
timestamp5 = 1544728504
dt_object1 = datetime.fromtimestamp(timestamp1)
dt_object2 = datetime.fromtimestamp(timestamp2)
dt_object3 = datetime.fromtimestamp(timestamp3)
dt_object4 = datetime.fromtimestamp(timestamp4)
dt_object5 = datetime.fromtimestamp(timestamp5)

print("dt_object =", dt_object1)
print("dt_object =", dt_object2)
print("dt_object =", dt_object3)
print("dt_object =", dt_object4)
print("dt_object =", dt_object5)
```

```
dt_object = 2018-12-16 09:30:08
dt_object = 2018-11-27 02:00:24
dt_object = 2018-12-03 06:28:03
dt_object = 2018-11-30 16:13:04
dt_object = 2018-12-13 19:15:04
```

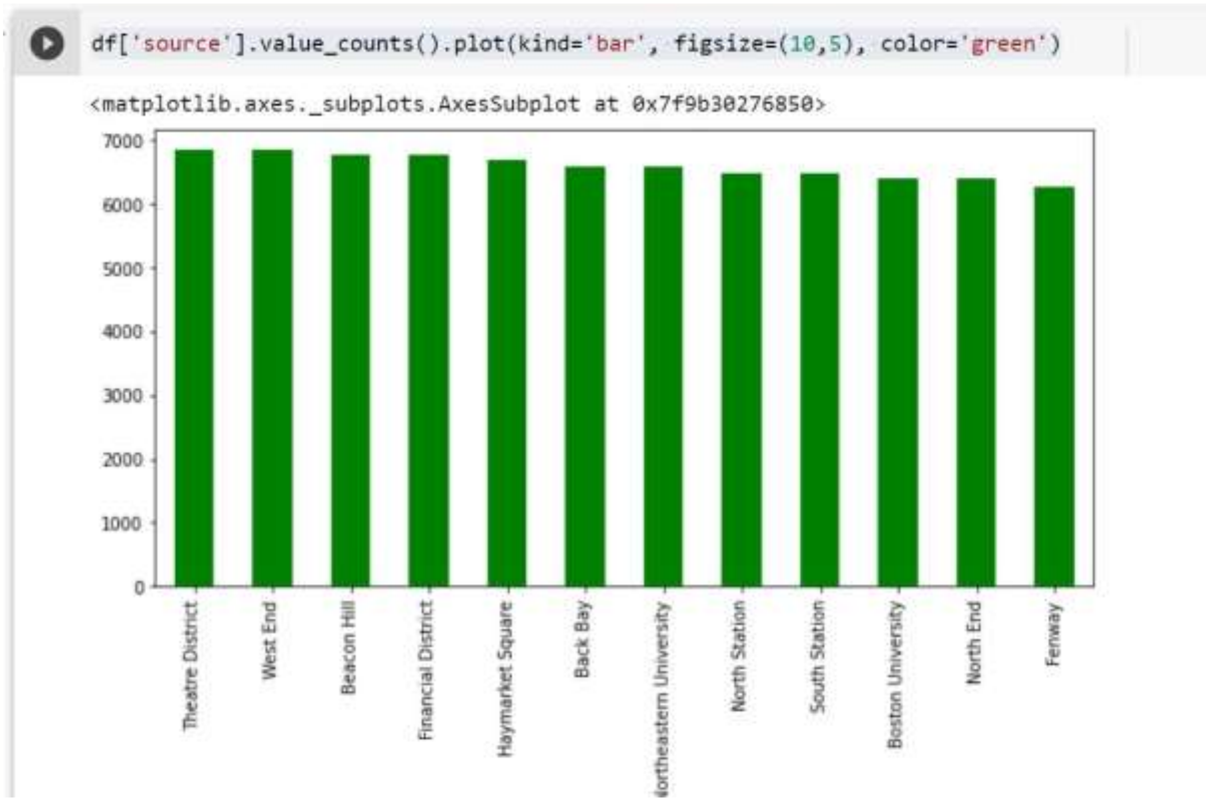
- So by this timestamp to datetime conversion we get to know that, our data is of the year 2018 and in the month of november and december only

BAR PLOTS:

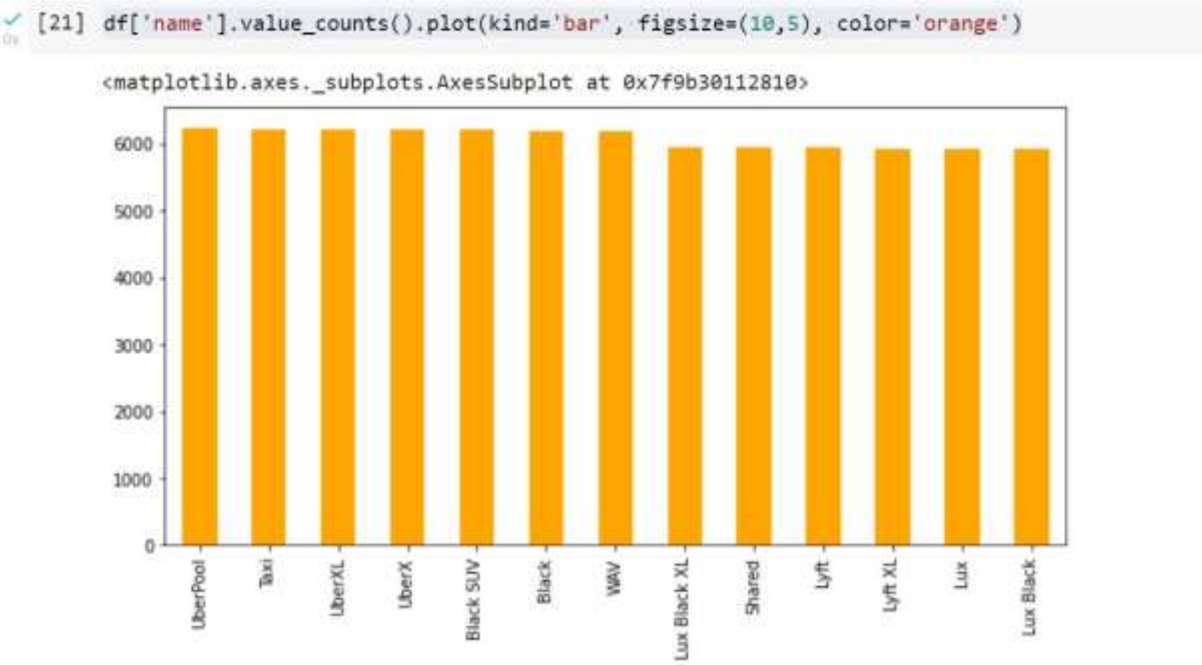
```
df['month'].value_counts().plot(kind='bar', figsize=(10,5), color='blue')
```



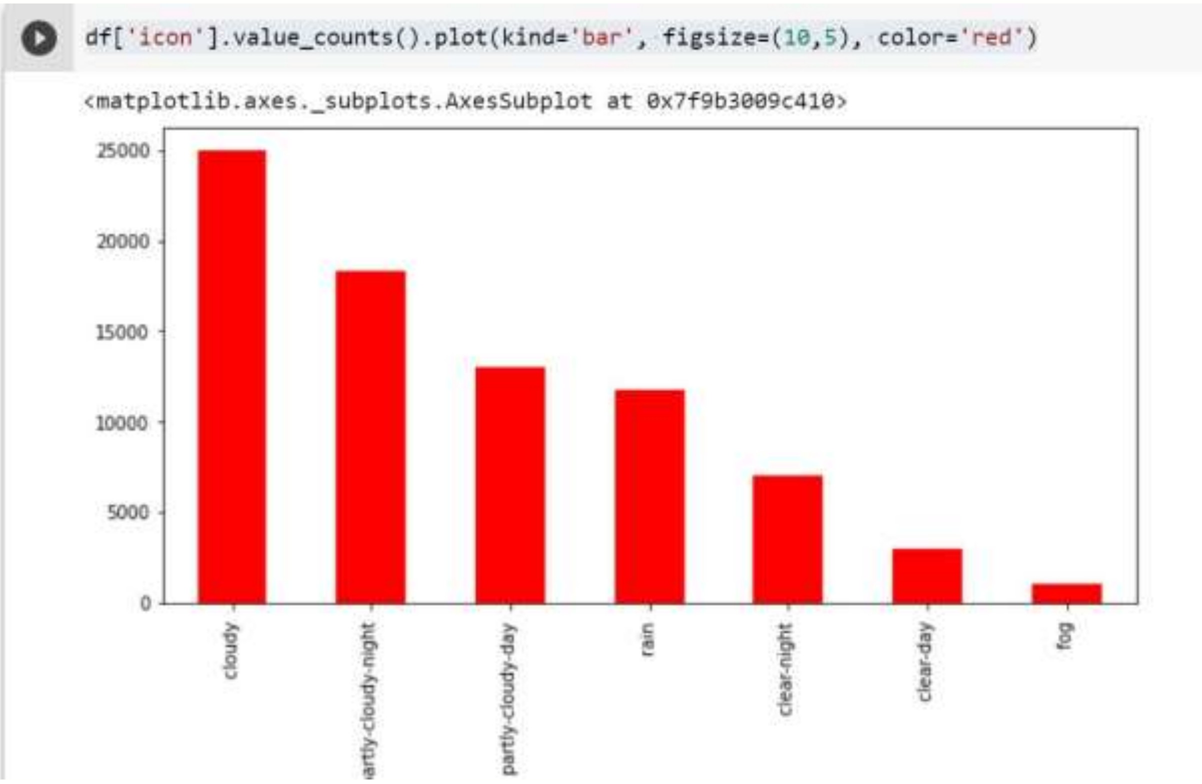

```
df['source'].value_counts().plot(kind='bar', figsize=(10,5), color='green')
```



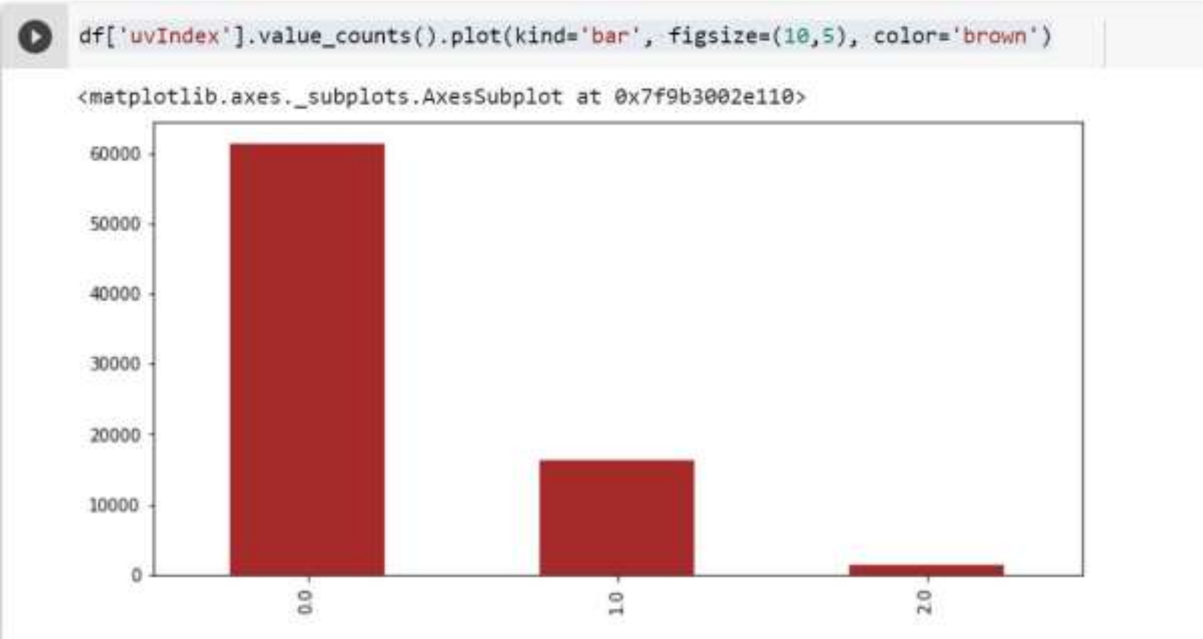
```
df['name'].value_counts().plot(kind='bar', figsize=(10,5), color='orange')
```



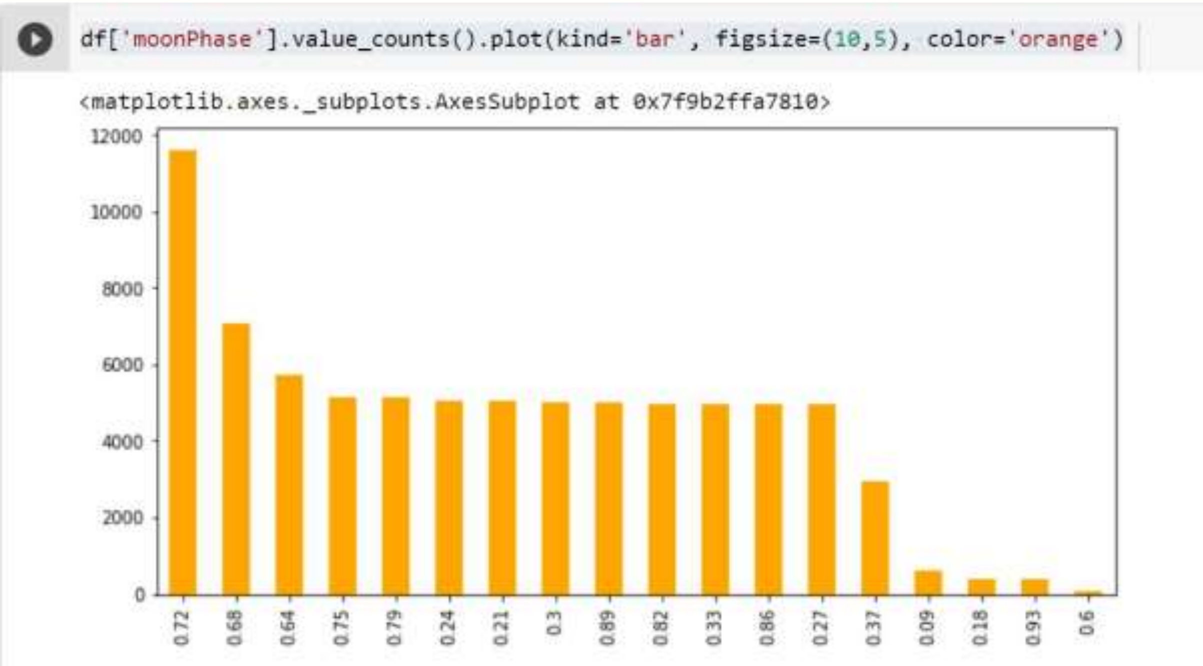

```
df['icon'].value_counts().plot(kind='bar', figsize=(10,5), color='red')
```



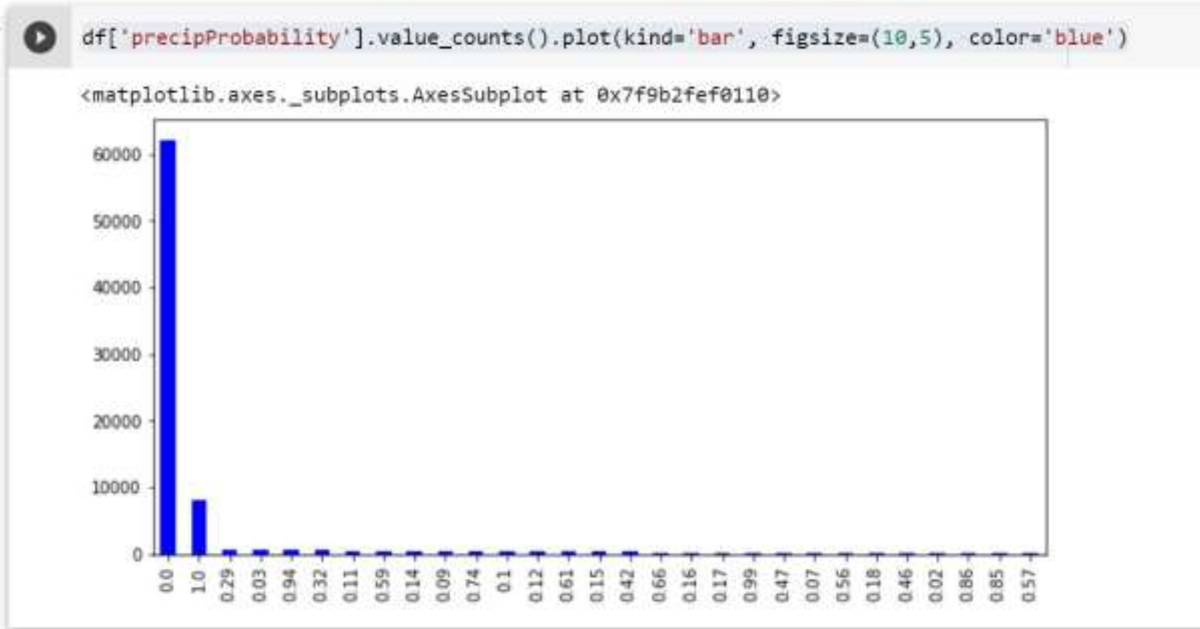
```
df['uvIndex'].value_counts().plot(kind='bar', figsize=(10,5), color='brown')
```




```
df['moonPhase'].value_counts().plot(kind='bar', figsize=(10,5), color='orange')
```



```
df['precipProbability'].value_counts().plot(kind='bar', figsize=(10,5), color='blue')
```



LABEL ENCODER:

```
# Import label encoder
from sklearn import preprocessing

# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()


df['id']= label_encoder.fit_transform(df['id'])

df['datetime']= label_encoder.fit_transform(df['datetime'])
df['timezone']= label_encoder.fit_transform(df['timezone'])
df['destination']= label_encoder.fit_transform(df['destination'])
df['product_id']= label_encoder.fit_transform(df['product_id'])
```

```
df['short_summary']= label_encoder.fit_transform(df['short_summary'])
df['long_summary']= label_encoder.fit_transform(df['long_summary'])


df['name']= label_encoder.fit_transform(df['name'])

print("Class mapping of Name: ")
for i, item in enumerate(label_encoder.classes_):
    print(item, "-->", i)
```

```
df['name']= label_encoder.fit_transform(df['name'])

print("Class mapping of Name: ")
for i, item in enumerate(label_encoder.classes_):
    print(item, "-->", i)
```

```
Class mapping of Name:
Black --> 0
Black SUV --> 1
Lux --> 2
Lux Black --> 3
Lux Black XL --> 4
Lyft --> 5
Lyft XL --> 6
Shared --> 7
Taxi --> 8
UberPool --> 9
UberX --> 10
UberXL --> 11
WAV --> 12
```

```
df['price'].median()
```

```
df['price'].median()
```

```
13.5
```

```
df["price"].fillna(10.5, inplace = True)
```

```
[40] df.isnull().sum()
```

```
id                      0
timestamp               0
hour                    0
day                     0
month                   0
datetime                0
timezone                0
source                  0
destination             0
cab_type                0
product_id              0
name                    0
price                   0
distance                0
surge_multiplier        0
latitude                0
longitude               0
temperature             0
apparentTemperature     0
short_summary           0
long_summary            0
precipIntensity         0
precipProbability       0
```

```
[41] df['price'].dtype

     dtype('float64')
```

```
[43] df['price'] = df['price'].astype(int)
```

```
df['price'].head()

0     5
1    11
2     7
3    26
4     9
Name: price, dtype: int64
```

RESULT ANALYSIS

After analyzing the various parameters, here are a few guidelines that we can conclude. If you were a Business analyst or data scientist working for **Uber or Lyft**, you could come to the following conclusions:

- Uber is very economical; however, Lyft also offers fair competition.
- People prefer to have a shared ride in the middle of the night.
- People avoid riding when it rains.
- When traveling long distances, the price does not increase by line. However, based on time and demand, increases can affect costs.
- Uber could be the first choice for long distances.

However, obtaining and analyzing the same data is the point of several companies. There are many businesses in the market that can help bring data from many sources and in various ways to your favorite data storage.

CONCLUSION

This project focuses on using machine-learning methods to predict the uber demand and visualize it in the shiny app. Shiny app provides an interactive way to understand the change of parameters, which is extremely helpful for selecting the models. Using this app, users can build the models as they want, and select the model with best performance