Market Segment Analysis of Online Vehicle Booking

by

Khushi Pathak

Dataset used:

https://drive.google.com/file/d/1SWSL4qCW5ZG3r3alXOGgOTre9SXiS4JO/view

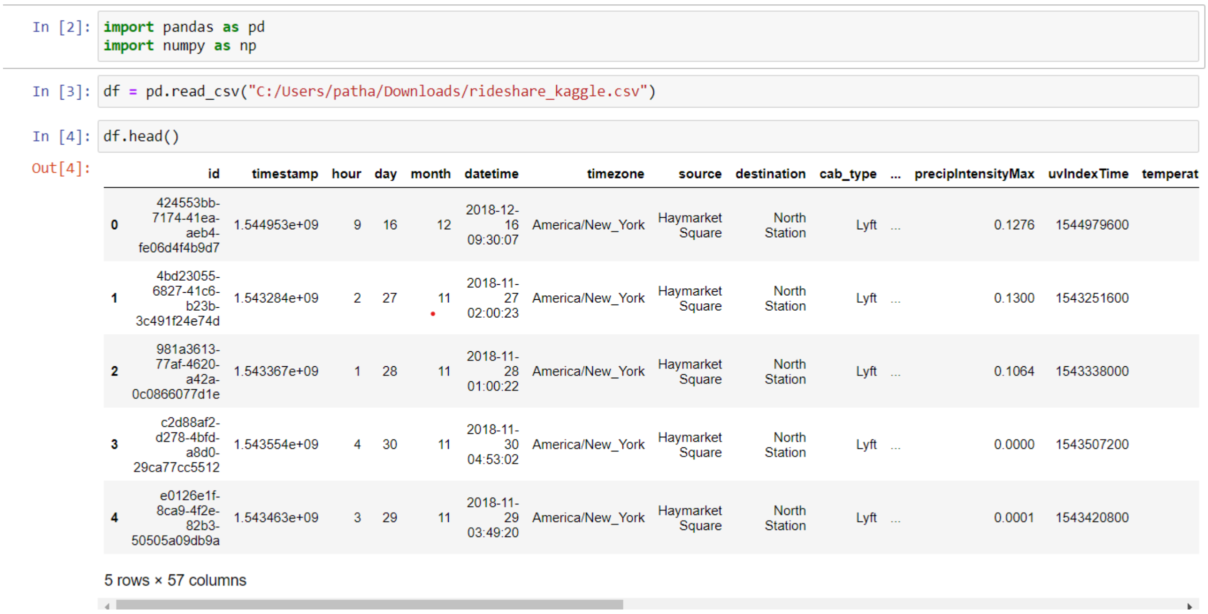
Project Link:

https://github.com/khushipathak3502/Feynn_labs/tree/main

1. Data Pre-Processing:

Data preprocessing is a crucial step in preparing raw data to make it suitable for machine learning models. The process involves cleaning the data, removing any errors or inconsistencies, and transforming it into a format that can be easily analyzed. It is essential to preprocess the data before performing any segmentation analysis.

To preprocess data, the first step is to import the raw data in a suitable format and create a data frame for further analysis. The next step is to identify any null values in the dataset and remove them to avoid any data inconsistencies.



```
In [2]: import pandas as pd
        import numpy as np
```

```
In [3]: df = pd.read_csv("C:/Users/patha/Downloads/rideshare_kaggle.csv")
```

```
In [4]: df.head()
```

Out[4]:

	id	timestamp	hour	day	month	datetime	timezone	source	destination	cab_type	...	precipIntensityMax	uvIndexTime	temperat
0	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	
1	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	
2	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	
3	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	
4	e0126e1f-8ca9-4f2e-82b3-50505a09db9a	1.543463e+09	3	29	11	2018-11-29 03:49:20	America/New_York	Haymarket Square	North Station	Lyft	...	0.0001	1543420800	

5 rows × 57 columns

```
In [6]: df.dropna(axis = 0, inplace = True)
        df.isnull().sum()
```

```
Out[6]: 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
        humidity                    0
        windSpeed                   0
        windGust                    0
        windGustTime                0
        visibility                  0
        temperatureHigh             0
        temperatureHighTime         0
        temperatureLow              0
```

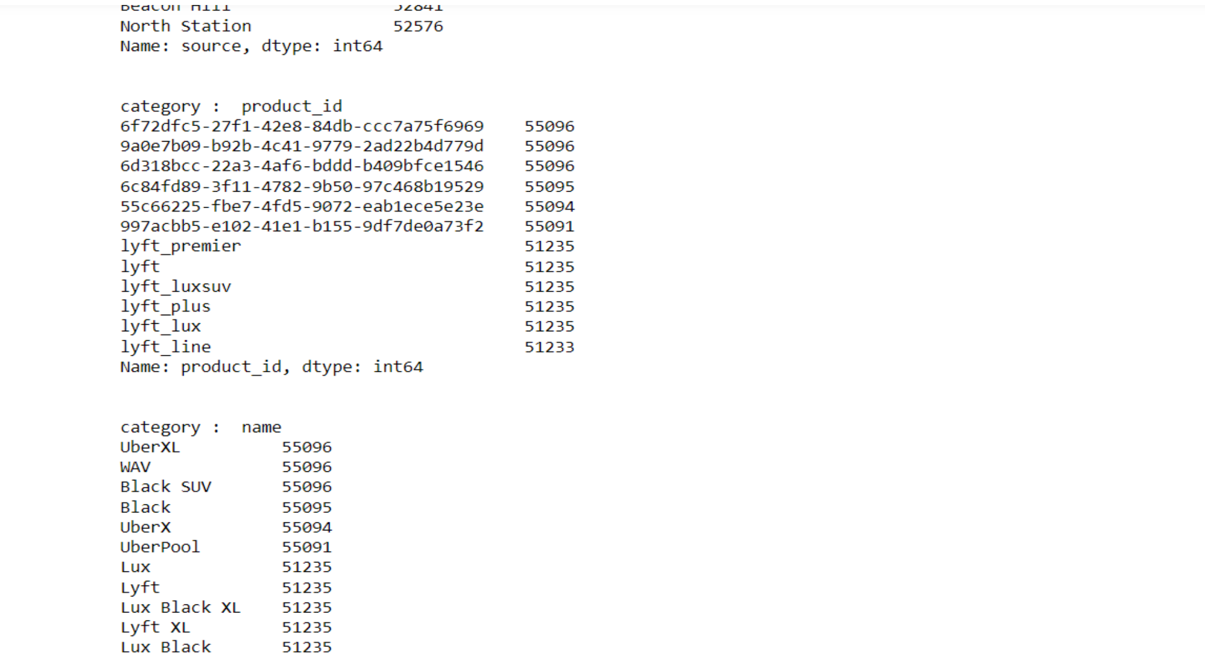
```
In [7]: selected =  df.loc[:, ["destination", "source", "product_id", "name"]]
```

```
In [8]: categorical = selected.select_dtypes('object').columns.tolist()
        categorical
```

```
Out[8]: ['destination', 'source', 'product_id', 'name']
```

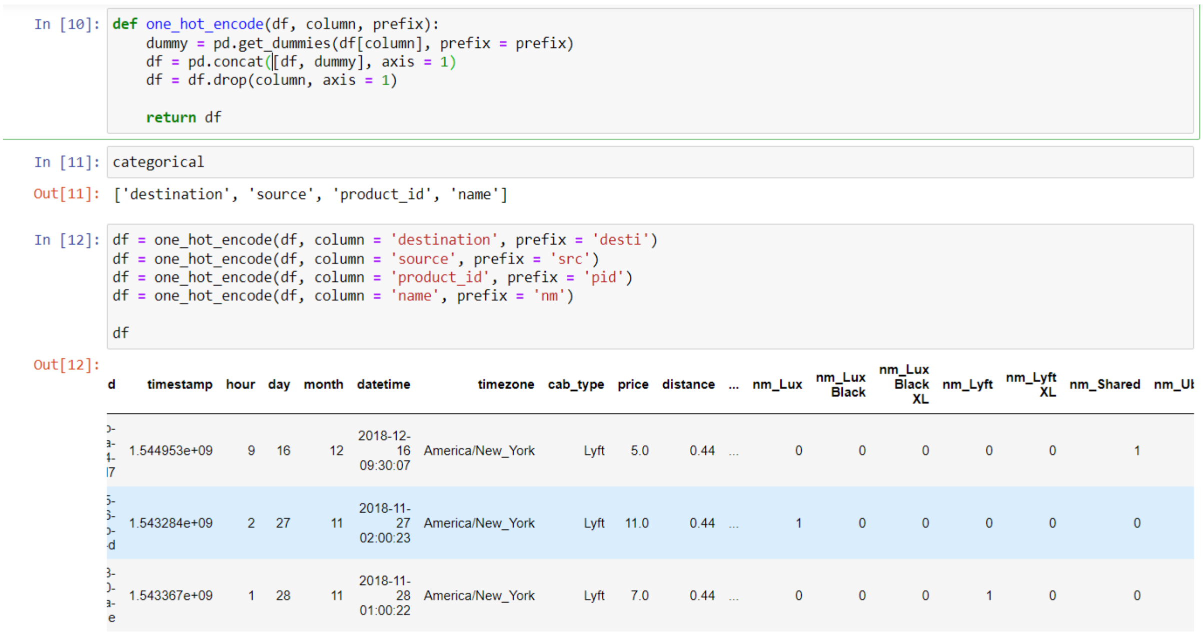
```
In [9]: for cat in categorical:
            print('category : ', cat)
            print(df[cat].value_counts())
            print('\n')
```

```
category :  destination
Financial District         54192
Back Bay                   53190
Theatre District           53189
Haymarket Square           53171
Boston University          53171
Fenway                     53166
Northeastern University    53165
North End                  53164
South Station              53159
West End                   52992
Beacon Hill                52840
North Station              52577
Name: destination, dtype: int64


category :  source
Financial District         54197
Back Bay                   53201
Theatre District           53201
Boston University          53172
```



```
Beacon Hill                 52841
North Station               52576
Name: source, dtype: int64


category :  product_id
6f72dfc5-27f1-42e8-84db-ccc7a75f6969    55096
9a0e7b09-b92b-4c41-9779-2ad22b4d779d    55096
6d318bcc-22a3-4af6-bddd-b409bfce1546    55096
6c84fd89-3f11-4782-9b50-97c468b19529    55095
55c66225-fbe7-4fd5-9072-eab1ece5e23e    55094
997acbb5-e102-41e1-b155-9df7de0a73f2    55091
lyft_premier                            51235
lyft                                    51235
lyft_luxsuv                             51235
lyft_plus                               51235
lyft_lux                                51235
lyft_line                               51233
Name: product_id, dtype: int64


category :  name
UberXL          55096
WAV             55096
Black SUV       55096
Black           55095
UberX           55094
UberPool        55091
Lux             51235
Lyft            51235
Lux Black XL    51235
Lyft XL         51235
Lux Black       51235
```

To make the attributes of data easier to understand we make changes to it known as One-hot encoding which is a technique used to represent categorical variables as numerical variables so that machine learning models can use them as inputs.



```
In [10]: def one_hot_encode(df, column, prefix):
             dummy = pd.get_dummies(df[column], prefix = prefix)
             df = pd.concat([df, dummy], axis = 1)
             df = df.drop(column, axis = 1)

             return df
```

```
In [11]: categorical
```

```
Out[11]: ['destination', 'source', 'product_id', 'name']
```

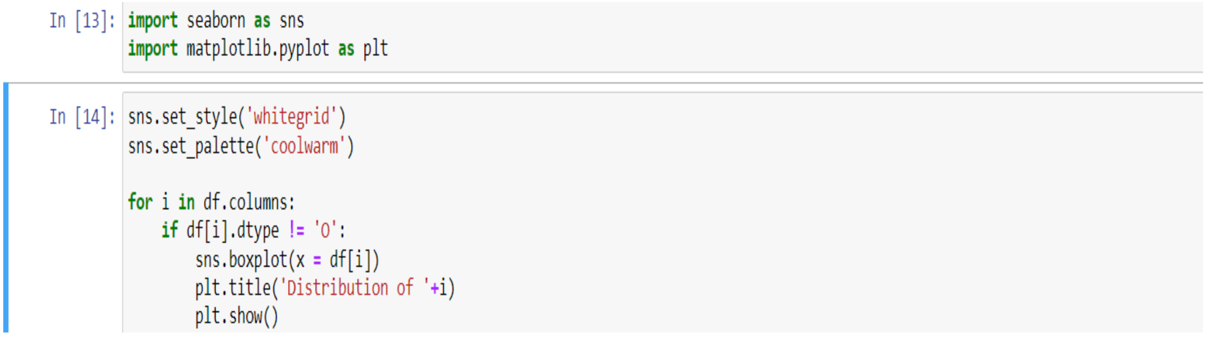
```
In [12]: df = one_hot_encode(df, column = 'destination', prefix = 'desti')
         df = one_hot_encode(df, column = 'source', prefix = 'src')
         df = one_hot_encode(df, column = 'product_id', prefix = 'pid')
         df = one_hot_encode(df, column = 'name', prefix = 'nm')

         df
```

Out[12]:

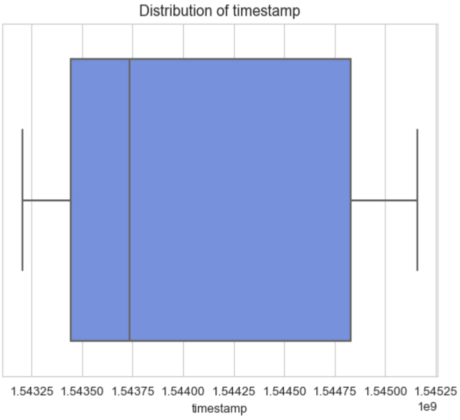
d	timestamp	hour	day	month	datetime	timezone	cab_type	price	distance	...	nm_Lux	nm_Lux Black	nm_Lux Black XL	nm_Lyft	nm_Lyft XL	nm_Shared	nm_Ut
)- a- 4- l7	1.544953e+09	9	16	12	2018-12-16 09:30:07	America/New_York	Lyft	5.0	0.44	...	0	0	0	0	0	1	
5- 3-)- -d	1.543284e+09	2	27	11	2018-11-27 02:00:23	America/New_York	Lyft	11.0	0.44	...	1	0	0	0	0	0	
3-)- a- e	1.543367e+09	1	28	11	2018-11-28 01:00:22	America/New_York	Lyft	7.0	0.44	...	0	0	0	1	0	0	

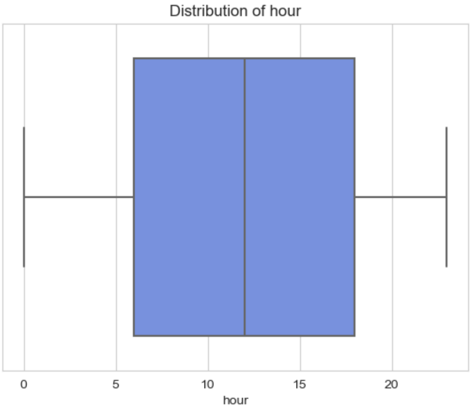
2. Visualization

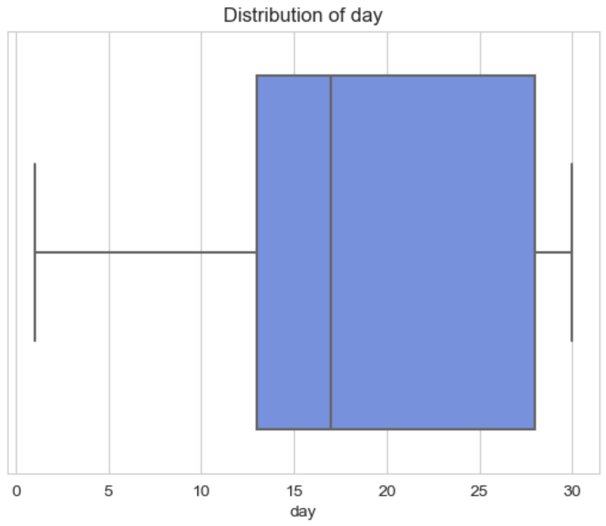
Data visualization is used to make complex data easier to understand, identify relationships and correlations, and communicate insights and findings to others. It also makes data more engaging, which can encourage people to explore it further. Finally, data visualization supports decision-making by providing a clear, visual representation of the data that can help identify trends and patterns that might be missed in other forms of analysis.

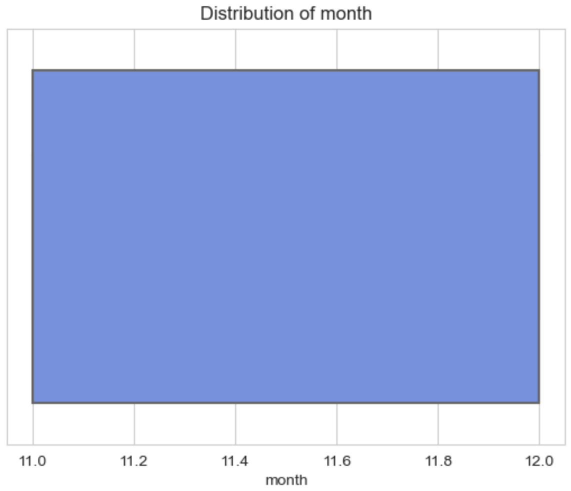


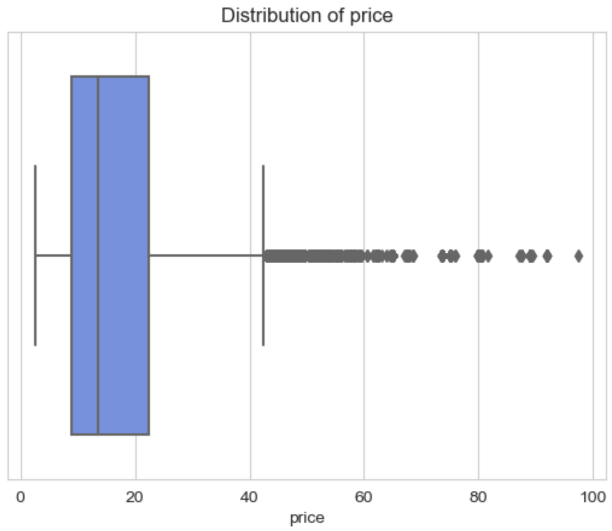
```
In [13]: import seaborn as sns
         import matplotlib.pyplot as plt

In [14]: sns.set_style('whitegrid')
         sns.set_palette('coolwarm')

         for i in df.columns:
             if df[i].dtype != 'O':
                 sns.boxplot(x = df[i])
                 plt.title('Distribution of '+i)
                 plt.show()
```

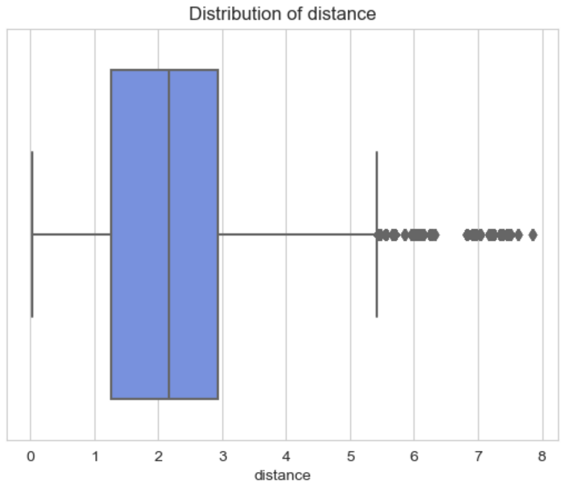


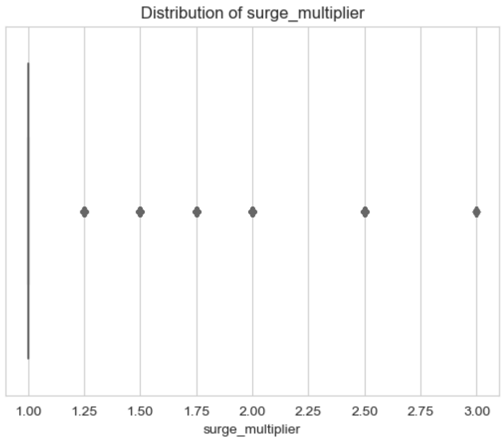


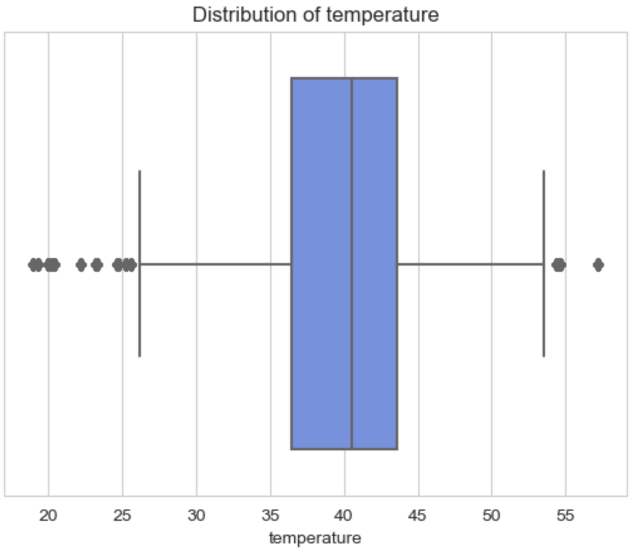

Distribution of day
0
5
10
15
20
25
30
day


Distribution of month
11.0
11.2
11.4
11.6
11.8
12.0
month









Observations:

The following are insights based on the given data:

- Data is almost equally distributed for source and destination:

The even distribution of data for source and destination suggests that the customers booking the cab are spread out across various locations, which can be advantageous for cab companies to optimize their services and coverage.

- Weather was cloudy when most of the customers booked cab and least of them booked on foggy day:

It appears that customers are more likely to book a cab on cloudy days and less likely on foggy days. This may be due to the fact that cloudy weather may not significantly impact transportation, while foggy weather can be challenging and even dangerous to drive in, causing people to avoid booking a cab altogether.

- Most of the cabs were booked in the midnight mostly after 10 P.M.:

The high number of bookings after 10 P.M. indicates that there is significant demand for transportation services during late hours. Cab companies can use this insight to optimize their services during these peak hours.

- Month end found to be the busiest days:

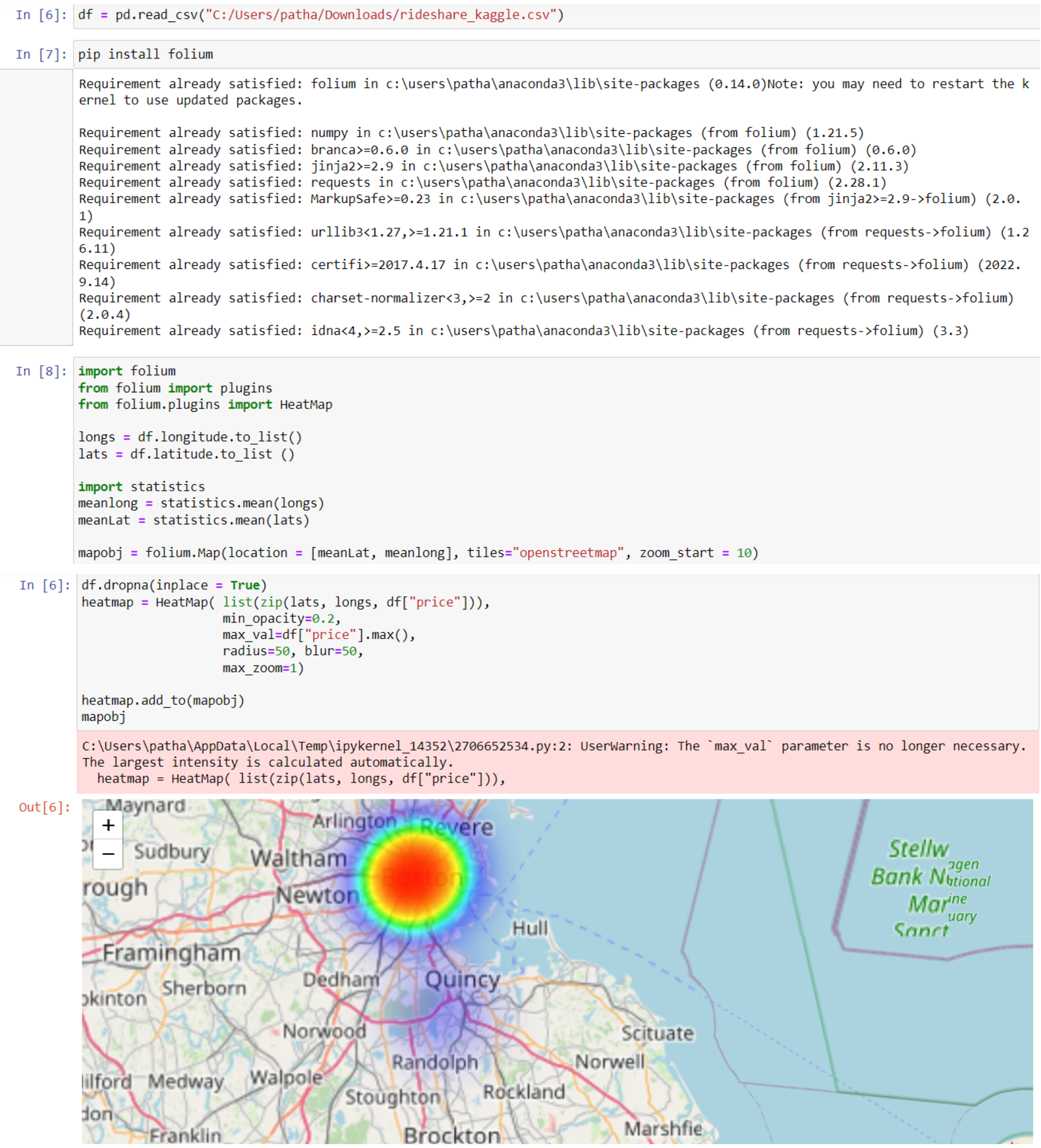
The observation that month-end is the busiest day for cab bookings could suggest that customers are more likely to book a cab during payday or while running errands to complete their month-end tasks.

- People mostly booking the cabs which are budget-friendly:

Customers preferring budget-friendly cabs could indicate that price sensitivity is an essential factor for cab bookings. Cab companies can use this insight to offer more budget-friendly services or pricing strategies to attract more customers.

3. Geometric Analysis

Geometric analysis is used to study geometric objects and their properties such as shape, size, and position. It is used to provide a rigorous mathematical foundation for various areas such as physics, engineering, and computer science. Geometric analysis enables the development of powerful tools to solve complex problems in these fields.



```
In [6]: df = pd.read_csv("C:/Users/patha/Downloads/rideshare_kaggle.csv")
```

```
In [7]: pip install folium
```

```
Requirement already satisfied: folium in c:\users\patha\anaconda3\lib\site-packages (0.14.0)Note: you may need to restart the k
ernel to use updated packages.

Requirement already satisfied: numpy in c:\users\patha\anaconda3\lib\site-packages (from folium) (1.21.5)
Requirement already satisfied: branca>=0.6.0 in c:\users\patha\anaconda3\lib\site-packages (from folium) (0.6.0)
Requirement already satisfied: jinja2>=2.9 in c:\users\patha\anaconda3\lib\site-packages (from folium) (2.11.3)
Requirement already satisfied: requests in c:\users\patha\anaconda3\lib\site-packages (from folium) (2.28.1)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\patha\anaconda3\lib\site-packages (from jinja2>=2.9->folium) (2.0.
1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\patha\anaconda3\lib\site-packages (from requests->folium) (1.2
6.11)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\patha\anaconda3\lib\site-packages (from requests->folium) (2022.
9.14)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\patha\anaconda3\lib\site-packages (from requests->folium)
(2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\patha\anaconda3\lib\site-packages (from requests->folium) (3.3)
```

```
In [8]: import folium
        from folium import plugins
        from folium.plugins import HeatMap

        longs = df.longitude.to_list()
        lats = df.latitude.to_list ()

        import statistics
        meanlong = statistics.mean(longs)
        meanLat = statistics.mean(lats)

        mapobj = folium.Map(location = [meanLat, meanlong], tiles="openstreetmap", zoom_start = 10)
```

```
In [6]: df.dropna(inplace = True)
        heatmap = HeatMap( list(zip(lats, longs, df["price"])),
                           min_opacity=0.2,
                           max_val=df["price"].max(),
                           radius=50, blur=50,
                           max_zoom=1)

        heatmap.add_to(mapobj)
        mapobj
```

```
C:\Users\patha\AppData\Local\Temp\ipykernel_14352\2706652534.py:2: UserWarning: The `max_val` parameter is no longer necessary.
The largest intensity is calculated automatically.
  heatmap = HeatMap( list(zip(lats, longs, df["price"])),
```

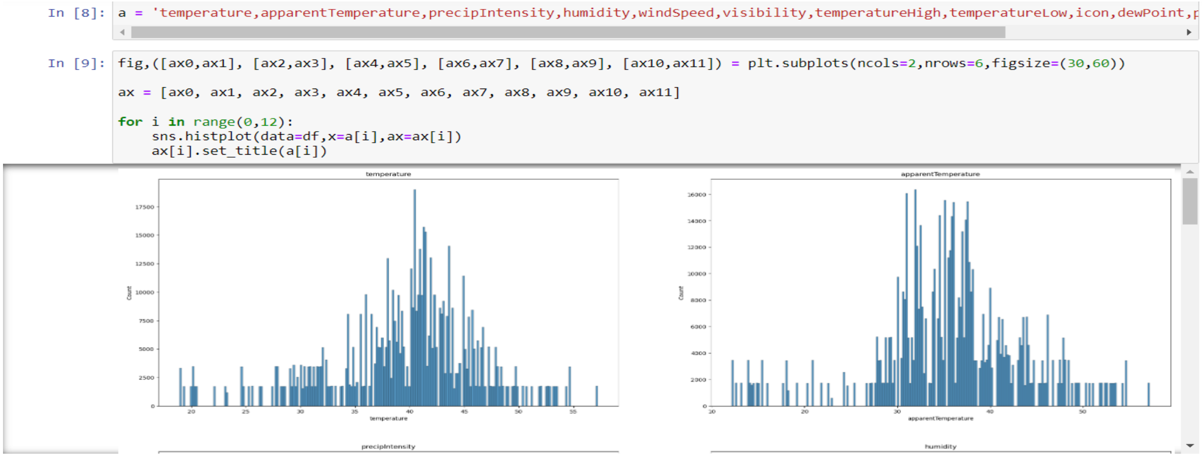
Out[6]:

Observations:

According to the available data, most of the cab bookings are from the Boston area. This could be due to various factors, such as a higher population density, greater business opportunities. Another possible factor could be the quality and reliability of cab services in the Boston area. Cab companies operating in this region may have a reputation for providing efficient and affordable transportation services.

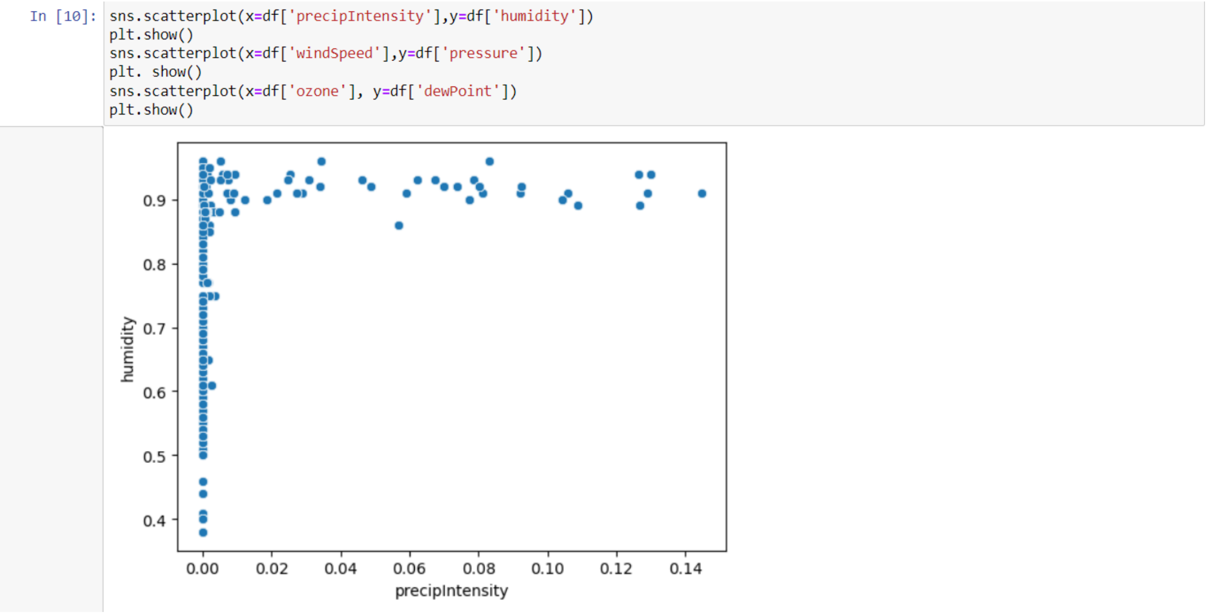
4. Psychographic Analysis

Psychographics helps in understanding consumer behaviour by analyzing their personality, values, interests, and lifestyle. It provides insights into the motivations and attitudes of the target audience, which can help marketers create more effective marketing strategies. By understanding the psychographics of their target audience, businesses can tailor their products and services to better meet customer needs and preferences.

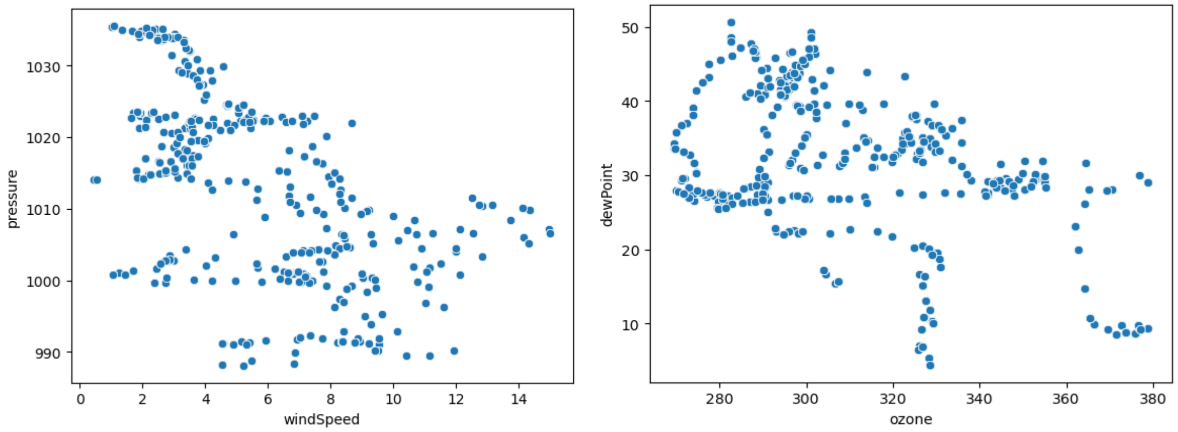


```
In [8]: a = 'temperature,apparentTemperature,precipIntensity,humidity,windSpeed,visibility,temperatureHigh,temperatureLow,icon,dewPoint,p
```

```
In [9]: fig,([ax0,ax1], [ax2,ax3], [ax4,ax5], [ax6,ax7], [ax8,ax9], [ax10,ax11]) = plt.subplots(ncols=2,nrows=6,figsize=(30,60))

        ax = [ax0, ax1, ax2, ax3, ax4, ax5, ax6, ax7, ax8, ax9, ax10, ax11]

        for i in range(0,12):
            sns.histplot(data=df,x=a[i],ax=ax[i])
            ax[i].set_title(a[i])
```



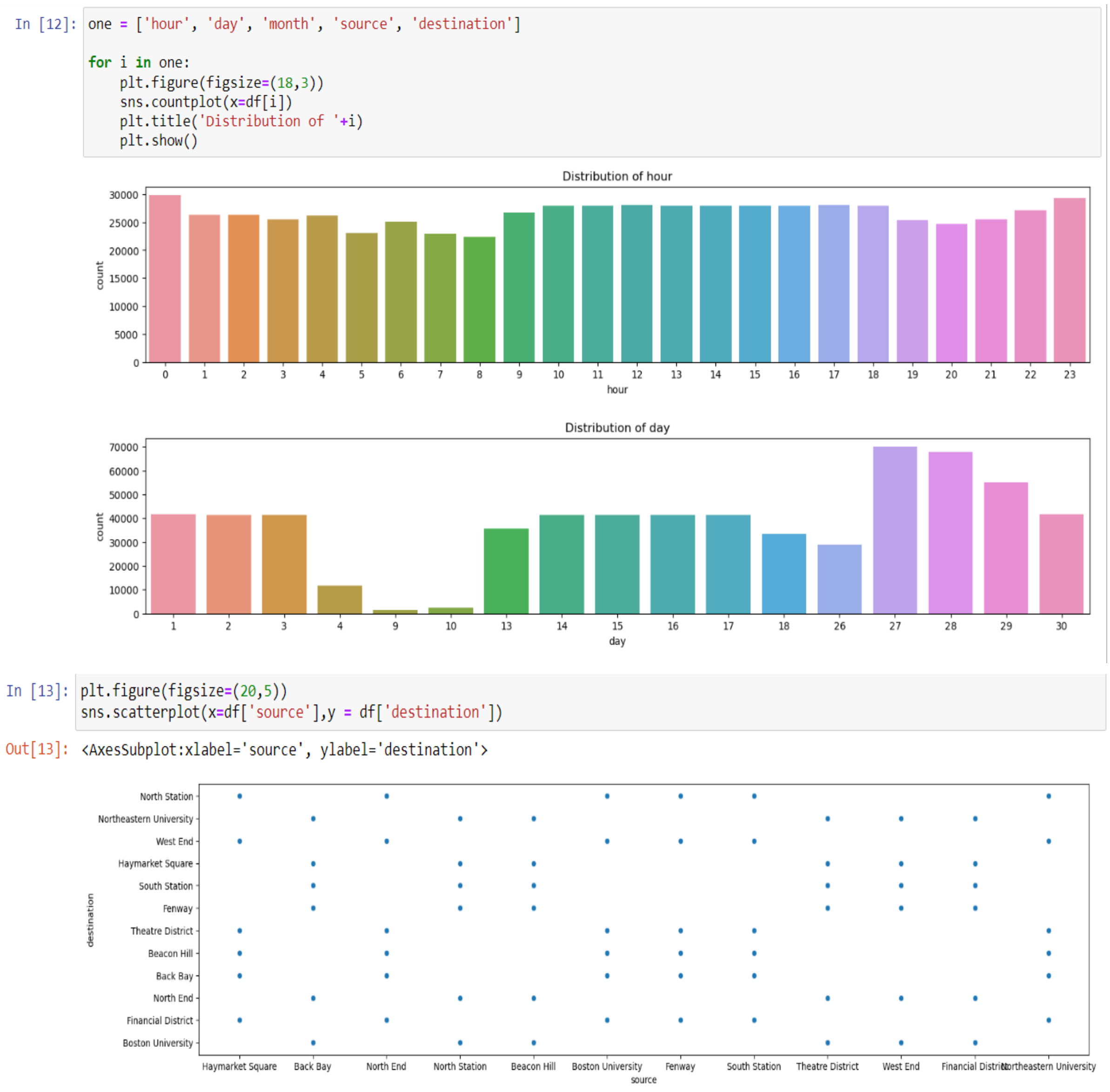
```
In [10]: sns.scatterplot(x=df['precipIntensity'],y=df['humidity'])
         plt.show()
         sns.scatterplot(x=df['windSpeed'],y=df['pressure'])
         plt. show()
         sns.scatterplot(x=df['ozone'], y=df['dewPoint'])
         plt.show()
```



Observations:
The distribution of temperature is approximately normal, with a majority of values ranging from 35 to 45 degrees. The preparation intensity is centered around 0.00, and visibility is around 10. Interestingly, the busiest day for cab bookings was found to be cloudy, while the least busy day was surprisingly a foggy day. Additionally, customers are more likely to ride a cab when there is a precipitation intensity greater than 0.01 and humidity greater than 0.8. These findings provide valuable insights into the relationships between weather conditions and cab booking behaviour.

5. Demographic Analysis:

Demographic analysis helps in understanding the characteristics of a population, such as age, gender, income, and education. It provides insights into the preferences and behaviors of a particular group, which can help in developing effective marketing strategies. By understanding the demographic makeup of their target audience, businesses can tailor their products and services to better meet customer needs and preferences.



```
In [12]: one = ['hour', 'day', 'month', 'source', 'destination']

         for i in one:
             plt.figure(figsize=(18,3))
             sns.countplot(x=df[i])
             plt.title('Distribution of '+i)
             plt.show()
```

```
In [13]: plt.figure(figsize=(20,5))
         sns.scatterplot(x=df['source'],y = df['destination'])
```

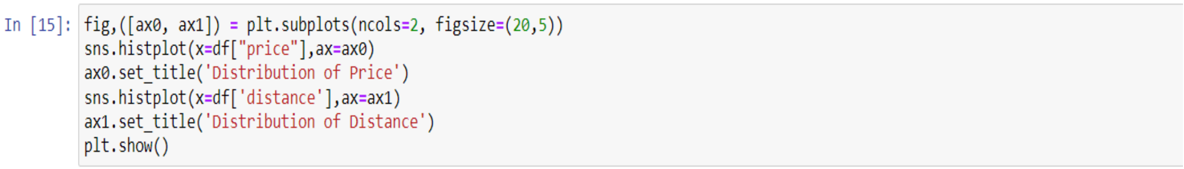
```
Out[13]: <AxesSubplot:xlabel='source', ylabel='destination'>
```

Observations:

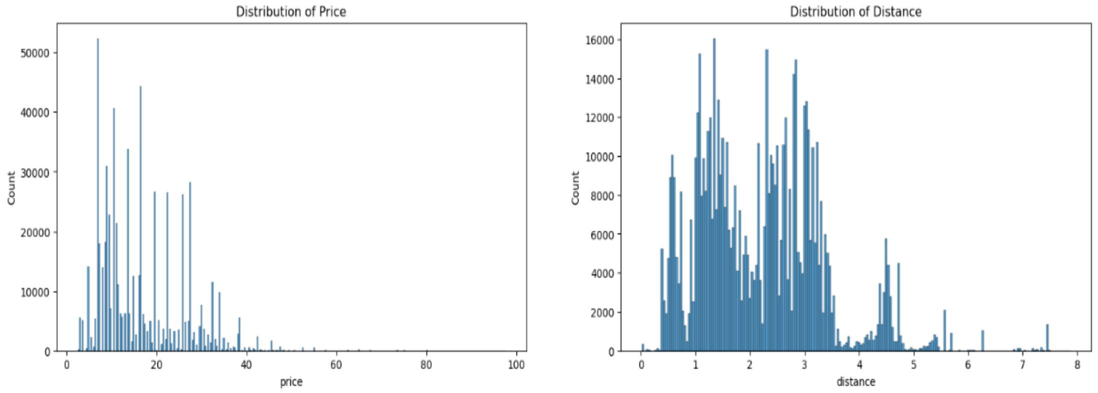
According to the data from the last two months of 2018, the busiest times for cab bookings are between 10 A.M. to 6 P.M. and after 10 P.M. Similarly, the end of the month appears to be the busiest period for cab drivers, while the period from the 4th to the 13th of each month sees relatively fewer cab bookings. The data also indicates that there is a relatively even distribution of cab bookings across all source and destination points.

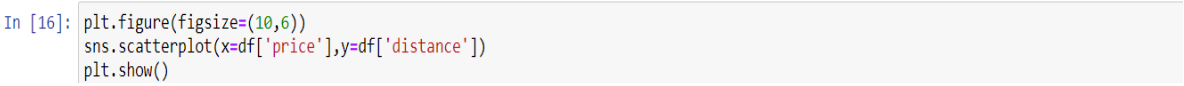
6. Behaviour Analysis:

Behaviour analysis helps in understanding the actions and choices made by individuals, providing insights into their preferences and motivations. It helps businesses identify the factors that influence consumer behaviour and develop effective marketing strategies. By understanding consumer behaviour, businesses can improve their products and services, enhance customer satisfaction, and increase profitability.

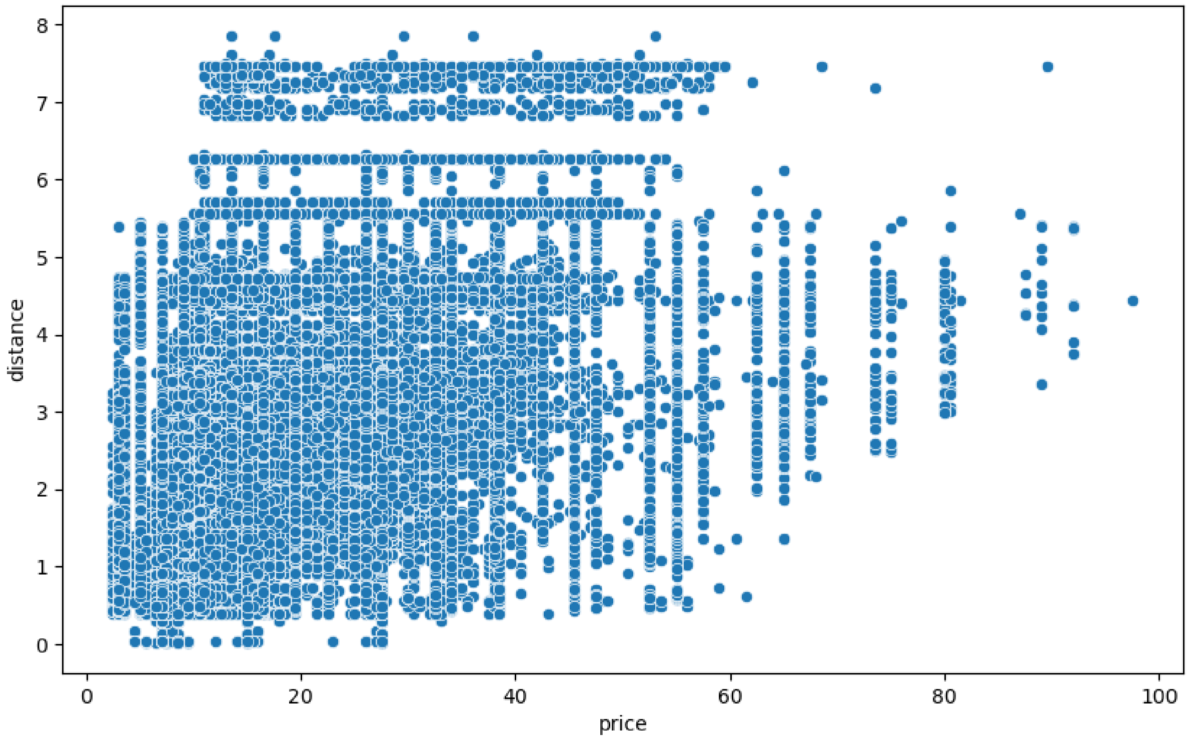


```
In [15]: fig,([ax0, ax1]) = plt.subplots(ncols=2, figsize=(20,5))
         sns.histplot(x=df["price"],ax=ax0)
         ax0.set_title('Distribution of Price')
         sns.histplot(x=df['distance'],ax=ax1)
         ax1.set_title('Distribution of Distance')
         plt.show()
```





```
In [16]: plt.figure(figsize=(10,6))
         sns.scatterplot(x=df['price'],y=df['distance'])
         plt.show()
```



Observations:

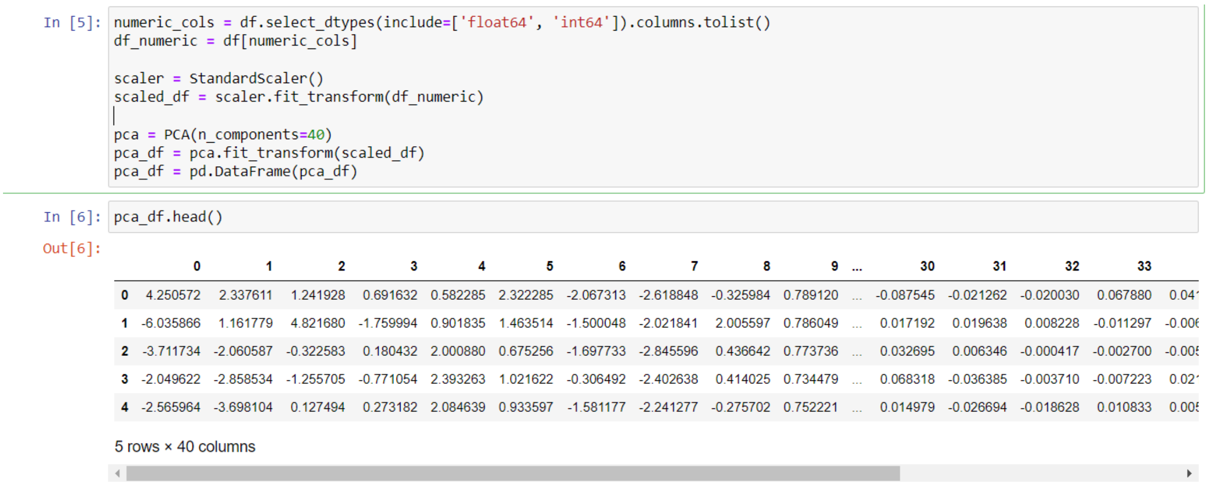
The majority of customers tend to book budget-friendly cabs with fares ranging from 5 to 25. Additionally, most customers prefer to book cabs for shorter distances, typically between 0.5 to 3.5 units. As the distance and price of the cab increases, the likelihood of customers booking decreases. These findings highlight the importance of price and distance in customer decision-making when it comes to booking a cab, and suggest that businesses should consider offering more affordable options for shorter trips to attract and retain customers.

Segment Extraction

Segment extraction is a process of dividing a larger population or market into smaller subgroups or segments based on certain criteria or characteristics such as demographics, psychographics, behaviour, and geographic location. This process allows businesses to understand their target audience in more detail and develop targeted marketing strategies to better meet their needs and preferences. By identifying distinct customer segments, businesses can tailor their products, services, and messaging to effectively reach and engage each group, which can lead to increased customer satisfaction and loyalty, as well as improved profitability. Overall, segment extraction is a crucial step in the marketing process that helps businesses optimize their resources and drive growth.

1. Clustering:

Clustering is important in identifying patterns and grouping similar data points together. It helps in understanding complex data sets, identifying market segments, and improving decision-making. Here by using K means clustering we have divided the set into 4 clusters using Elbow Curve method.



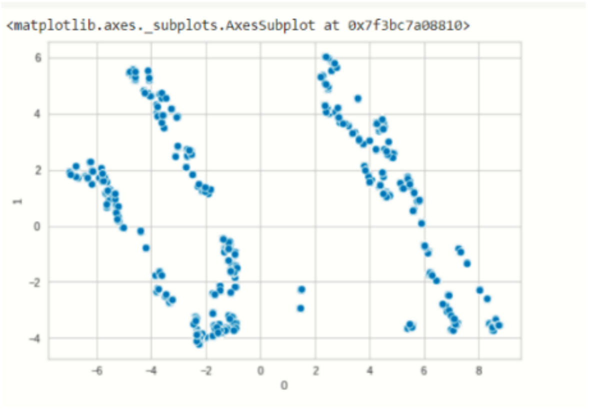
```
In [5]: numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns.tolist()
        df_numeric = df[numeric_cols]

        scaler = StandardScaler()
        scaled_df = scaler.fit_transform(df_numeric)

        pca = PCA(n_components=40)
        pca_df = pca.fit_transform(scaled_df)
        pca_df = pd.DataFrame(pca_df)
```

```
In [6]: pca_df.head()
```

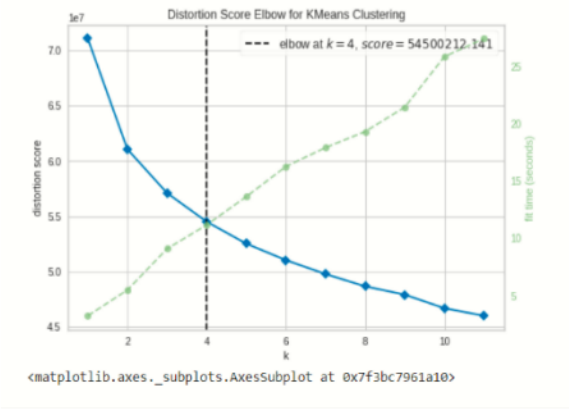
Out[6]:

	0	1	2	3	4	5	6	7	8	9	...	30	31	32	33	
0	4.250572	2.337611	1.241928	0.691632	0.582285	2.322285	-2.067313	-2.618848	-0.325984	0.789120	...	-0.087545	-0.021262	-0.020030	0.067880	0.04
1	-6.035866	1.161779	4.821680	-1.759994	0.901835	1.463514	-1.500048	-2.021841	2.005597	0.786049	...	0.017192	0.019638	0.008228	-0.011297	-0.006
2	-3.711734	-2.060587	-0.322583	0.180432	2.000880	0.675256	-1.697733	-2.845596	0.436642	0.773736	...	0.032695	0.006346	-0.000417	-0.002700	-0.005
3	-2.049622	-2.858534	-1.255705	-0.771054	2.393263	1.021622	-0.306492	-2.402638	0.414025	0.734479	...	0.068318	-0.036385	-0.003710	-0.007223	0.02
4	-2.565964	-3.698104	0.127494	0.273182	2.084639	0.933597	-1.581177	-2.241277	-0.275702	0.752221	...	0.014979	-0.026694	-0.018628	0.010833	0.005

5 rows × 40 columns

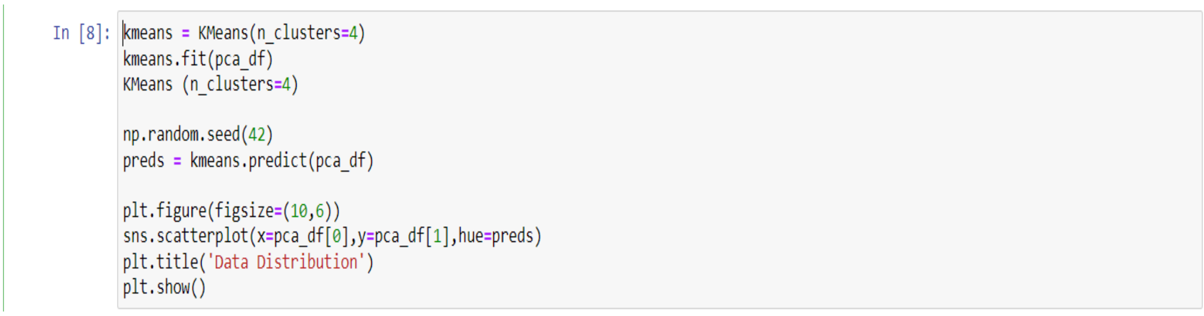


```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3bc7a08810>
```

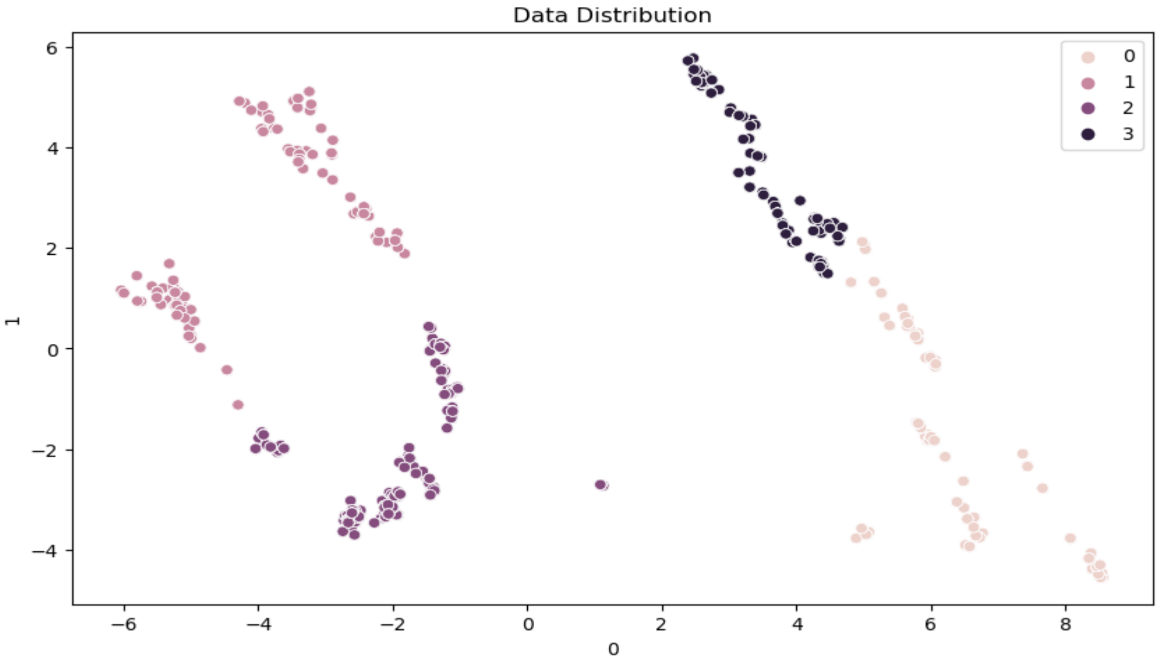


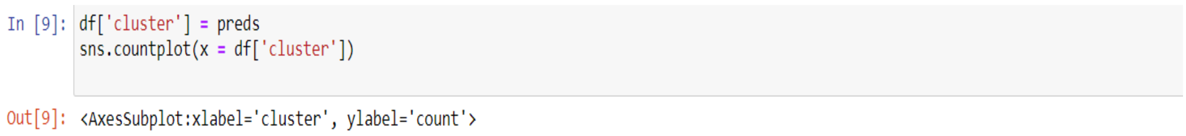
Distortion Score Elbow for KMeans Clustering

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3bc7961a10>
```

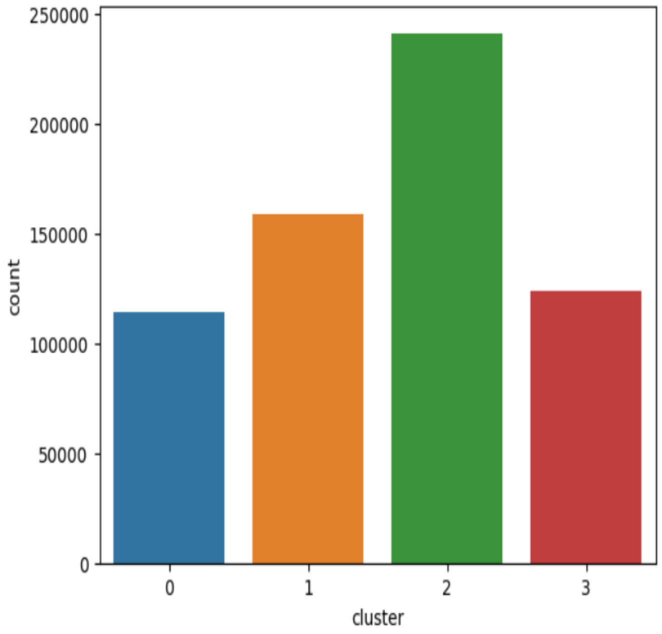


```
In [8]: kmeans = KMeans(n_clusters=4)
        kmeans.fit(pca_df)
        KMeans (n_clusters=4)

        np.random.seed(42)
        preds = kmeans.predict(pca_df)

        plt.figure(figsize=(10,6))
        sns.scatterplot(x=pca_df[0],y=pca_df[1],hue=preds)
        plt.title('Data Distribution')
        plt.show()
```



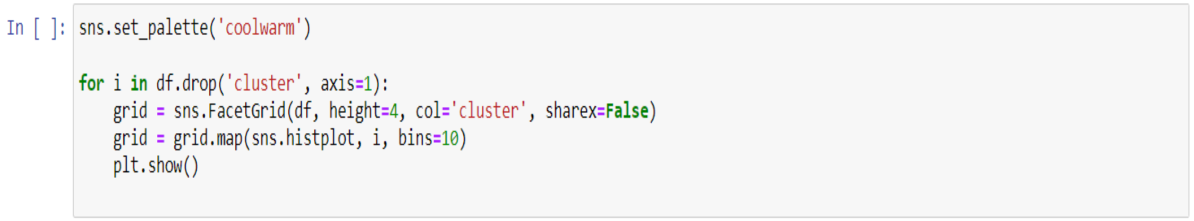


```
In [9]: df['cluster'] = preds
        sns.countplot(x = df['cluster'])

Out[9]: <AxesSubplot:xlabel='cluster', ylabel='count'>
```

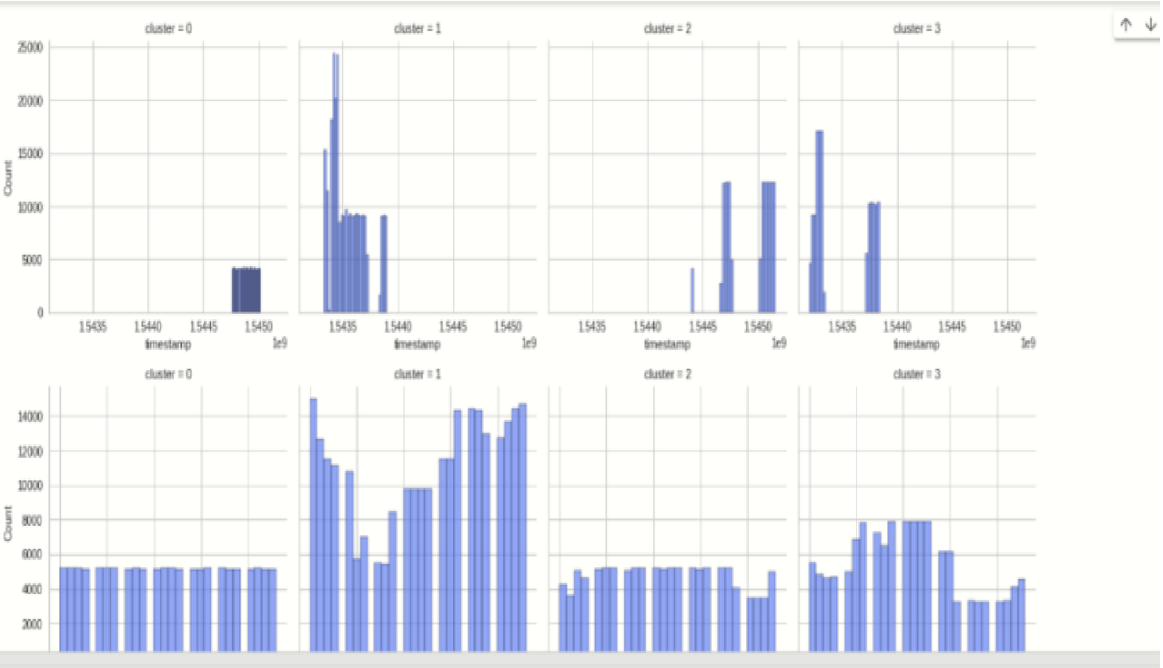


2. Profiling Segments:

Profiling segments involves analyzing the characteristics and behaviours of each segment identified through clustering. This analysis includes demographic and psychographic factors, as well as consumer behaviour patterns, such as purchase history and preferences. Profiling helps businesses understand the unique needs and preferences of each segment, allowing them to tailor their marketing strategies to effectively engage and meet the needs of each group.



```
In [ ]: sns.set_palette('coolwarm')

        for i in df.drop('cluster', axis=1):
            grid = sns.FacetGrid(df, height=4, col='cluster', sharex=False)
            grid = grid.map(sns.histplot, i, bins=10)
            plt.show()
```



Final conclusions

Based on the analysis of the cab booking data from the last two months of 2018, we can draw several observations and conclusions:

1. The majority of customers prefer to book budget-friendly cabs, with fares ranging from 5 to 25, for shorter distances ranging from 0.5 to 3.5 units.
2. Customers are more likely to book a cab during the day, with the busiest times being between 10 A.M. to 6 P.M. and after 10 P.M.
3. The end of the month is the busiest period for cab drivers, while the period from the 4th to the 13th of each month sees relatively fewer cab bookings.
4. The data shows an even distribution of cab bookings across all source and destination points.
5. The temperature distribution is approximately normal, with a majority of values ranging from 35 to 45 degrees.

6. The busiest day for cab bookings was found to be cloudy, while the least busy day was a foggy day.
7. Customers are more likely to ride a cab when there is a precipitation intensity greater than 0.01 and humidity greater than 0.8.
8. Most of the cab bookings are from the Boston area, indicating a high demand for efficient and affordable transportation services in the region.
9. Price sensitivity is an essential factor for cab bookings, with customers preferring budget-friendly options.
10. Cab companies can use these insights to optimize their services, pricing strategies, and coverage to attract and retain more customers.

To start a new cab service, various parameters should be considered, including customer demographics, booking behaviour, weather conditions, and time of day. For example, offering budget-friendly options for shorter distances could attract customers who prioritize affordability. It is also important to consider the busiest times for cab bookings, such as late evenings and month-end periods, and optimize services accordingly. Furthermore, monitoring weather conditions and understanding the relationship between weather and cab bookings can help in predicting demand and optimizing services accordingly. By considering these factors, a new cab service can tailor its services to meet customer needs and preferences and potentially gain a competitive advantage in the market.