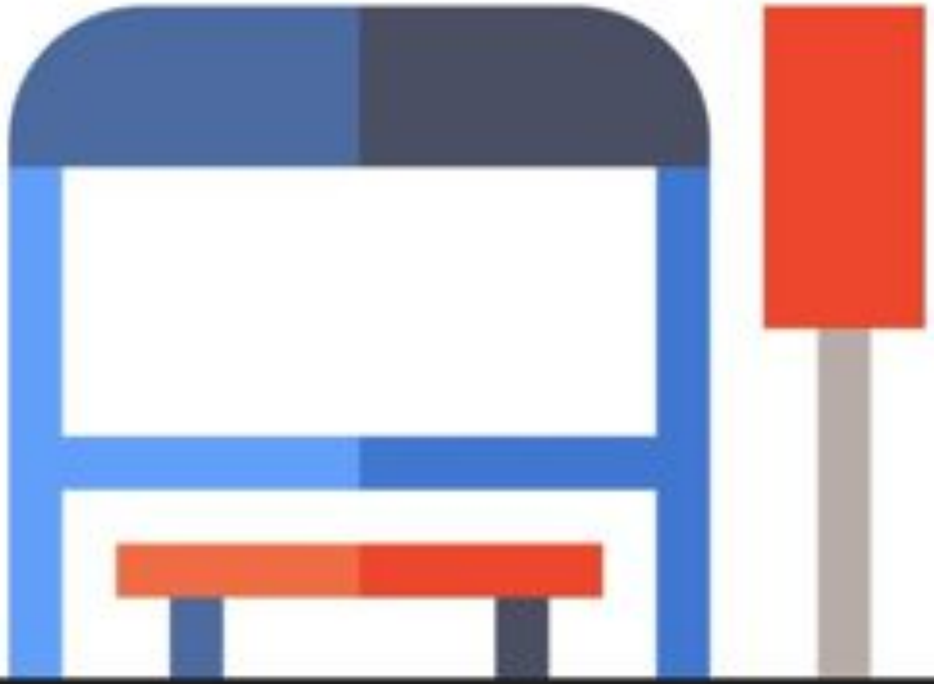
Behavior of Ride Sharing in Distinct Boston Areas and External Influences

...How To Make Money as a Rideshare Driver in Boston

Jill, Rick, & Tova

Summary

- Ride sharing has become an increasingly important component of city transportation options.
- It is estimated that Uber and Lyft drivers take ~ $20 million dollars away from the MBTA in Boston
- Usage of ridesharing increased 25 percent between 2017 and 2018, with 81.3 million trips.



Motivation

Given that usage has been increasing over time and provides a viable occupation, as a team we want to analyze data from ride share companies to understand where in Boston rides are generated and under what circumstances in order to help drivers maximize their likelihood of a pickup, and therefore profit.

SO MANY QUESTIONS!!!

There are so many questions we could answer with the large data set we were working with. Ultimately, our goal is to help rideshare drivers find the "perfect" conditions in which to find a rider.

Below are some of the basic questions, along with the type of data required to answer them.

01

What area of Boston are most rides sourced from?

(Geographical Data)

02

What time of day are most rides requested?

(Time Stamp Data)

03

What day of the week are most rides requested?
(Calendar Data)

04

At what apparent temperature are the most rides requested?
(Weather Data)

Data Cleanup & Exploration

- We found a data set using Kaggle that had the majority of information we were looking for.
- Used weather and geography APIs for heatmaps.
- All NaNs were removed.
- Graphing correlations were sparse.
- We double checked the data several times during coding to ensure we were graphing variables correctly.
- Dataset was slightly unclear on defining some of it’s variables.
- Dataset was very large, and we trimmed it from 57 to 31 columns to consider.

Summary of the Dataset

In [96]:
```
#What are the mean state of the factors in consideration?
mean_hour = clean_ride_data["hour"].mean()
mean_price = clean_ride_data["price"].mean()
mean_distance = clean_ride_data["distance"].mean()
mean_surge_muliplier = clean_ride_data["surge_multiplier"].mean()
mean_temperature = clean_ride_data["temperature"].mean()
mean_apparentTemperature = clean_ride_data["apparentTemperature"].mean()
mean_windSpeed = clean_ride_data["windSpeed"].mean()
mean_visibility = clean_ride_data["visibility"].mean()
mean_precip = clean_ride_data["precipProbability"].mean()

# Create a new Datatframe for summary stats

Summary_table = {
                 "Average Time of Day":[mean_hour],
                 "Average Price": [mean_price],
                 "Average Ride Distance": [mean_distance],
                 "Average Surge Multiplier":[mean_surge_muliplier],
                 "Average Temperature":[mean_temperature],
                 "Average Apparent Temperature": [mean_apparentTemperature],
                 "Average Windspeed": [mean_windSpeed],
                 "Average Visibility":[mean_visibility],
                 "Average Precip":[mean_precip]

                }

# TO DO, if we like this, we can continue to format all the data in the summary table
Summary_table_df = pd.DataFrame(Summary_table)
Summary_table_df['Average Time of Day'] = Summary_table_df['Average Time of Day'].round(decimals=1)
Summary_table_df['Average Price'] = Summary_table_df['Average Price'].map('${:,.2f}'.format)
Summary_table_df['Average Ride Distance'] = Summary_table_df['Average Ride Distance'].round(decimals=1)
Summary_table_df['Average Surge Multiplier'] = Summary_table_df['Average Surge Multiplier'].round(decimals=1)
Summary_table_df['Average Temperature'] = Summary_table_df['Average Temperature'].round(decimals=1)
Summary_table_df['Average Apparent Temperature'] = Summary_table_df['Average Apparent Temperature'].round(decimals=1)
Summary_table_df['Average Windspeed'] = Summary_table_df['Average Windspeed'].round(decimals=1)
Summary_table_df['Average Visibility'] = Summary_table_df['Average Visibility'].round(decimals=1)
Summary_table_df['Average Precip'] = Summary_table_df['Average Precip'].round(decimals=3)

Summary_table_df
```

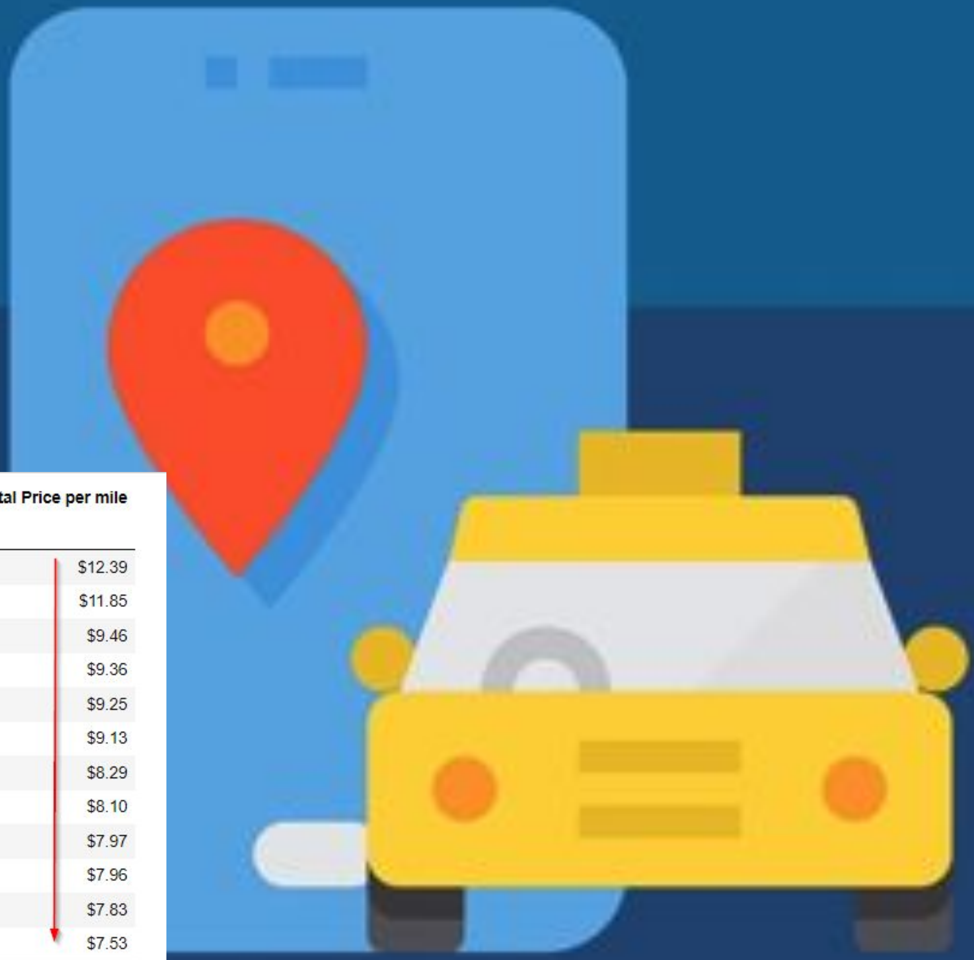
Out[96]:

	Average Time of Day	Average Price	Average Ride Distance	Average Surge Multiplier	Average Temperature	Average Apparent Temperature	Average Windspeed	Average Visibility	Average Precip
0	11.6	$16.55	2.2	1.0	39.6	35.9	6.2	8.5	0.146

Grouped Analysis

```python
grouped_Stats_df = clean_ride_data.groupby(['cab_type','source'])
Grouped_mean_df = grouped_Stats_df['hour'].mean()
Grouped_price_df =  grouped_Stats_df['price'].mean()
Grouped_distance_df = grouped_Stats_df['distance'].mean()
Mean_cost = Grouped_price_df/Grouped_distance_df
Number_rides = grouped_Stats_df['source'].count()

Summary_table = {
            "Total Source Counts": Number_rides,
            "Mean Time of Day": Grouped_mean_df,
            "Mean Price": Grouped_price_df,
            "Mean Distance": Grouped_distance_df,
            "Mean Total Price per mile": Mean_cost
}

Summary_table_df = pd.DataFrame(Summary_table)

Summary_table_df['Mean Time of Day'] = Summary_table_df['Mean Time of Day'].round(decimals=1)
Summary_table_df['Mean Price'] = Summary_table_df['Mean Price'].map('${:,.2f}'.format)
Summary_table_df['Mean Distance'] = Summary_table_df['Mean Distance'].round(decimals=1)


#Sort the table based on parameter of interest and then convert it to a price
Summary_table_Sort_df = Summary_table_df.sort_values("Mean Total Price per mile", ascending=False)
Summary_table_Sort_df['Mean Total Price per mile'] = Summary_table_Sort_df['Mean Total Price per mile'].map('${:,.2f}'.format

Summary_table_Sort_df
```

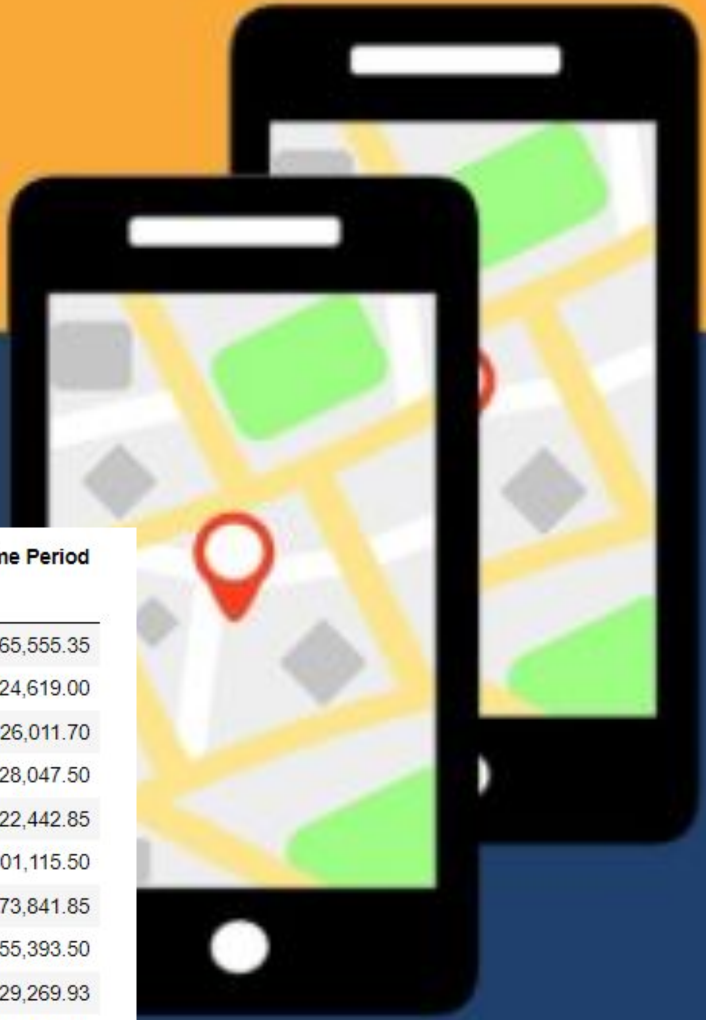
cab_type	source	Total Source Counts	Mean Time of Day	Mean Price	Mean Distance	Mean Total Price per mile
Uber	Haymarket Square	32122	11.5	$13.43	1.1	$12.39
Lyft	Haymarket Square	25614	11.6	$13.74	1.2	$11.85
	North End	25620	11.6	$15.62	1.7	$9.46
Uber	North End	32143	11.7	$14.72	1.6	$9.36
Lyft	Back Bay	25655	11.6	$16.56	1.8	$9.25
	South Station	25620	11.6	$16.30	1.8	$9.13
Uber	Theatre District	32283	11.7	$15.02	1.8	$8.29
Lyft	Theatre District	25530	11.6	$18.31	2.3	$8.10
	Beacon Hill	25464	11.5	$16.40	2.1	$7.97
Uber	South Station	32130	11.7	$15.08	1.9	$7.96
Lyft	West End	25488	11.6	$16.69	2.1	$7.83
	North Station	25326	11.8	$16.96	2.3	$7.53
Uber	Beacon Hill	31939	11.5	$14.98	2.0	$7.44
	West End	32074	11.6	$15.57	2.1	$7.30
	North Station	31792	11.6	$15.81	2.2	$7.23
Lyft	Northeastern University	25614	11.7	$19.02	2.6	$7.20

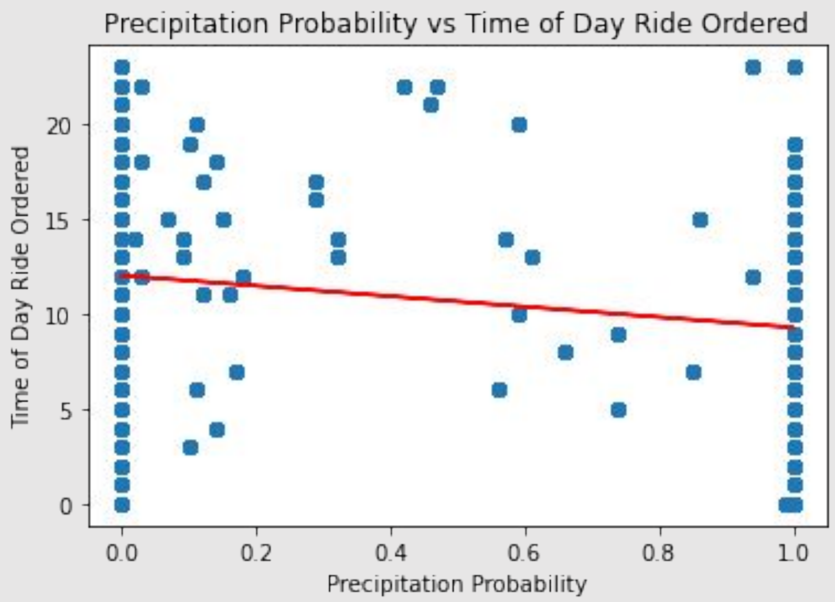


Binned Analysis

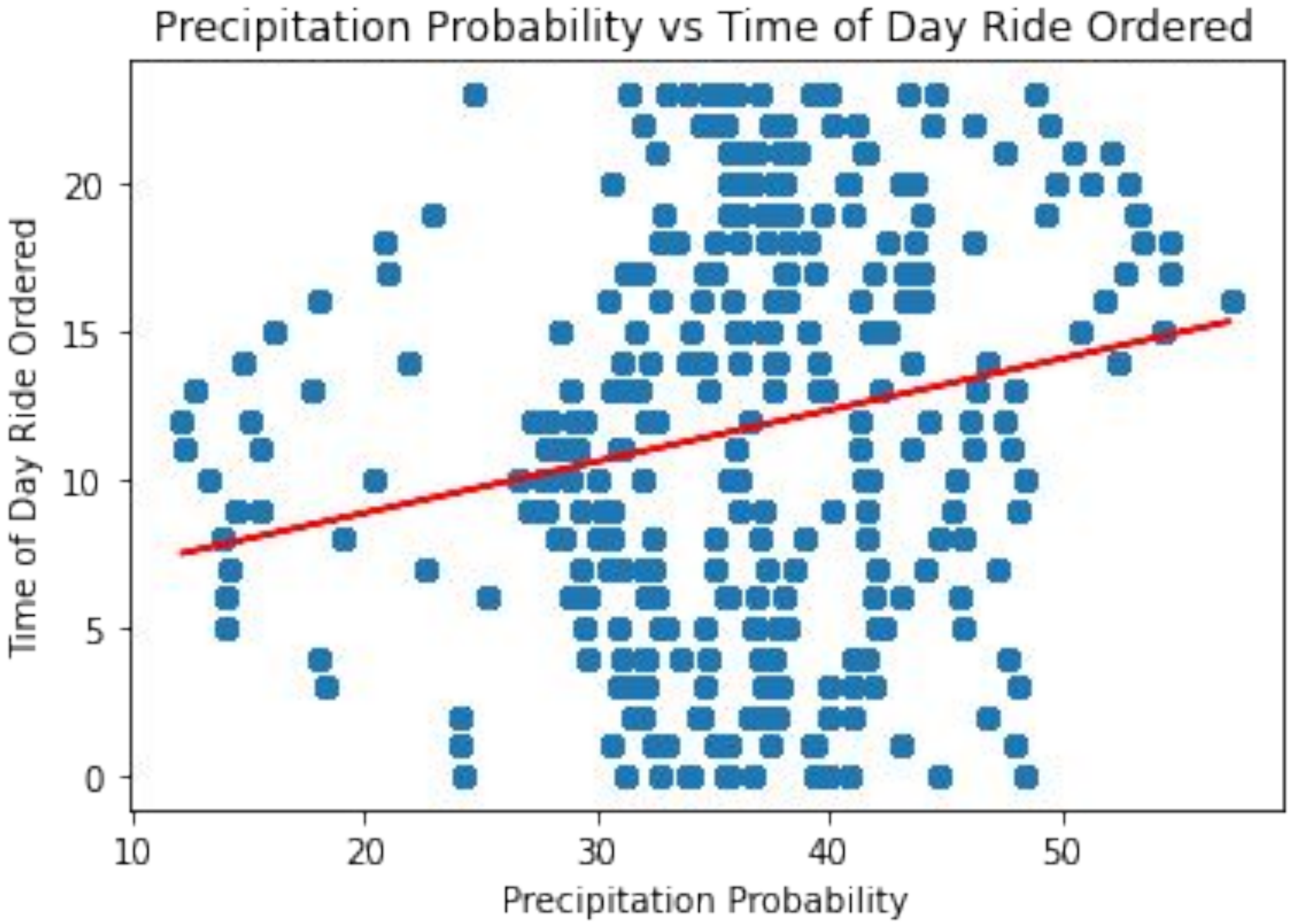
```
In [98]:
# Deep dive on time of day versus count using binning
# Set Bins
bins = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24]
group_names = ["Earlier than 2 am", ">2-4 am", ">4-6 am", ">6-8 am", ">8-10 am", ">10 am to noon", ">12-2 pm", ">2-4 pm", ">4

clean_ride_data["Time of Day of Ride"] = pd.cut(clean_ride_data["hour"], bins, labels=group_names, include_lowest=True)

# Creating a group based off of the bins
Hour_group_df = clean_ride_data.groupby("Time of Day of Ride")

# Find how many rows fall into each bin
Total_rides_by_time = (Hour_group_df["source"].count())
Average_price_by_time = (Hour_group_df["price"].mean())
Average_distance_by_time = (Hour_group_df["distance"].mean())
Average_price_by_distance = Average_price_by_time/Average_distance_by_time
Total_price_by_time = (Hour_group_df["price"].sum())
Total_price_by_time = (Hour_group_df["price"].sum())


BinnedHours = {
                         "Total Count": Total_rides_by_time,
                         "Average Price": Average_price_by_time,
                         "Average Price per Mile": Average_price_by_distance,
                         "Total Value of Trips for Time Period": Total_price_by_time

                      }

BinnedHours_df= pd.DataFrame(BinnedHours)

BinnedHours_Sort_df = BinnedHours_df.sort_values("T
BinnedHours_Sort_df["Average Price per Mile"] = Bin
BinnedHours_Sort_df['Average Price'] = BinnedHours_
BinnedHours_Sort_df['Total Value of Trips for Time


BinnedHours_Sort_df
```

Time of Day of Ride	Total Count	Average Price	Average Price per Mile	Total Value of Trips for Time Period
Earlier than 2 am	89509	$16.55	$7.57	$1,365,555.35
>10 am to noon	60768	$16.52	$7.57	$924,619.00
>12-2 pm	60768	$16.55	$7.57	$926,011.70
>4-6 pm	60768	$16.56	$7.53	$928,047.50
>2-4 pm	60767	$16.52	$7.54	$922,442.85
>8-10 am	59355	$16.51	$7.51	$901,115.50
>8-10 pm	57168	$16.60	$7.56	$873,841.85
>2-4 am	56145	$16.56	$7.59	$855,393.50
>6-8 pm	54337	$16.58	$7.53	$829,269.93
>4-6 am	52344	$16.53	$7.56	$795,684.80
>6-8 am	49211	$16.55	$7.57	$749,021.50
>10 pm - midnight	31931	$16.50	$7.59	$484,389.50

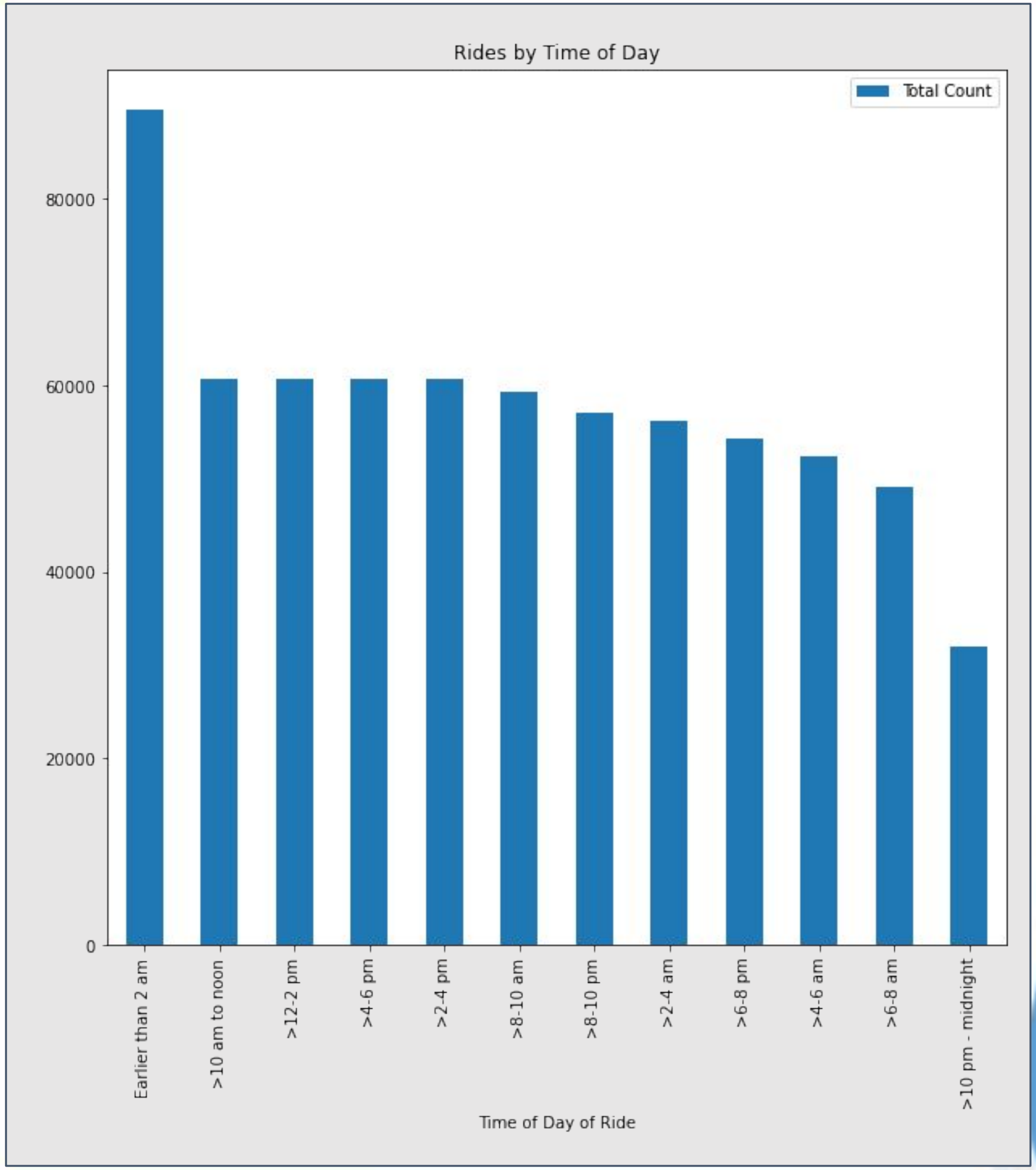


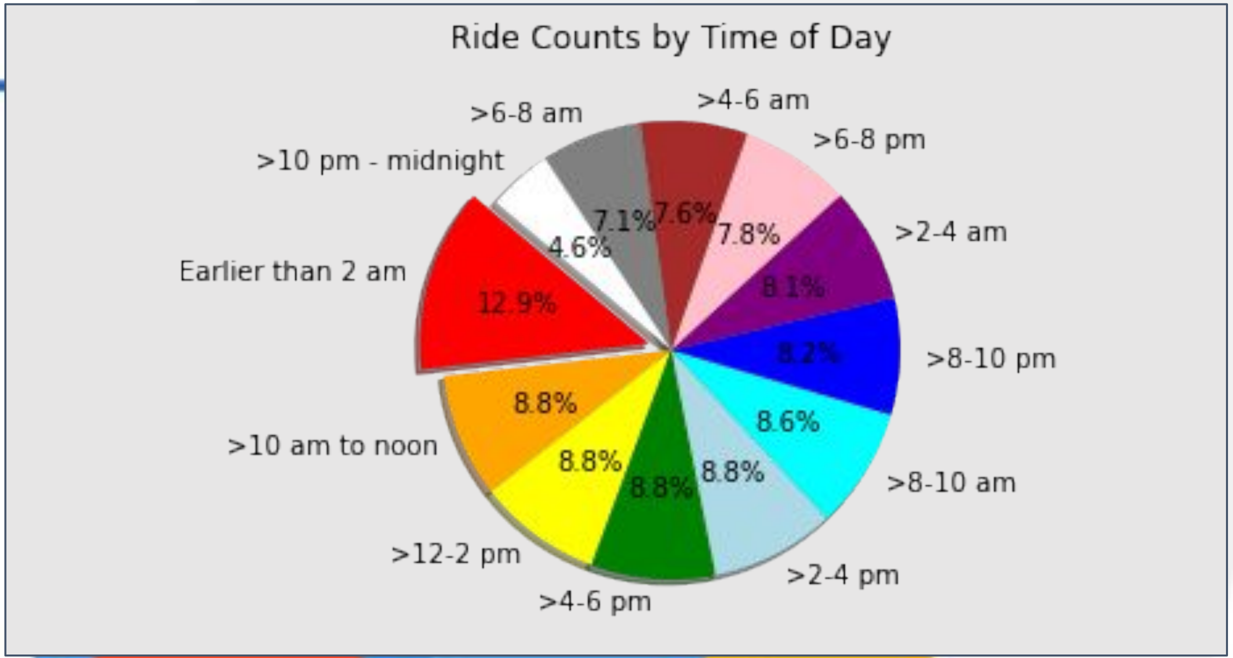


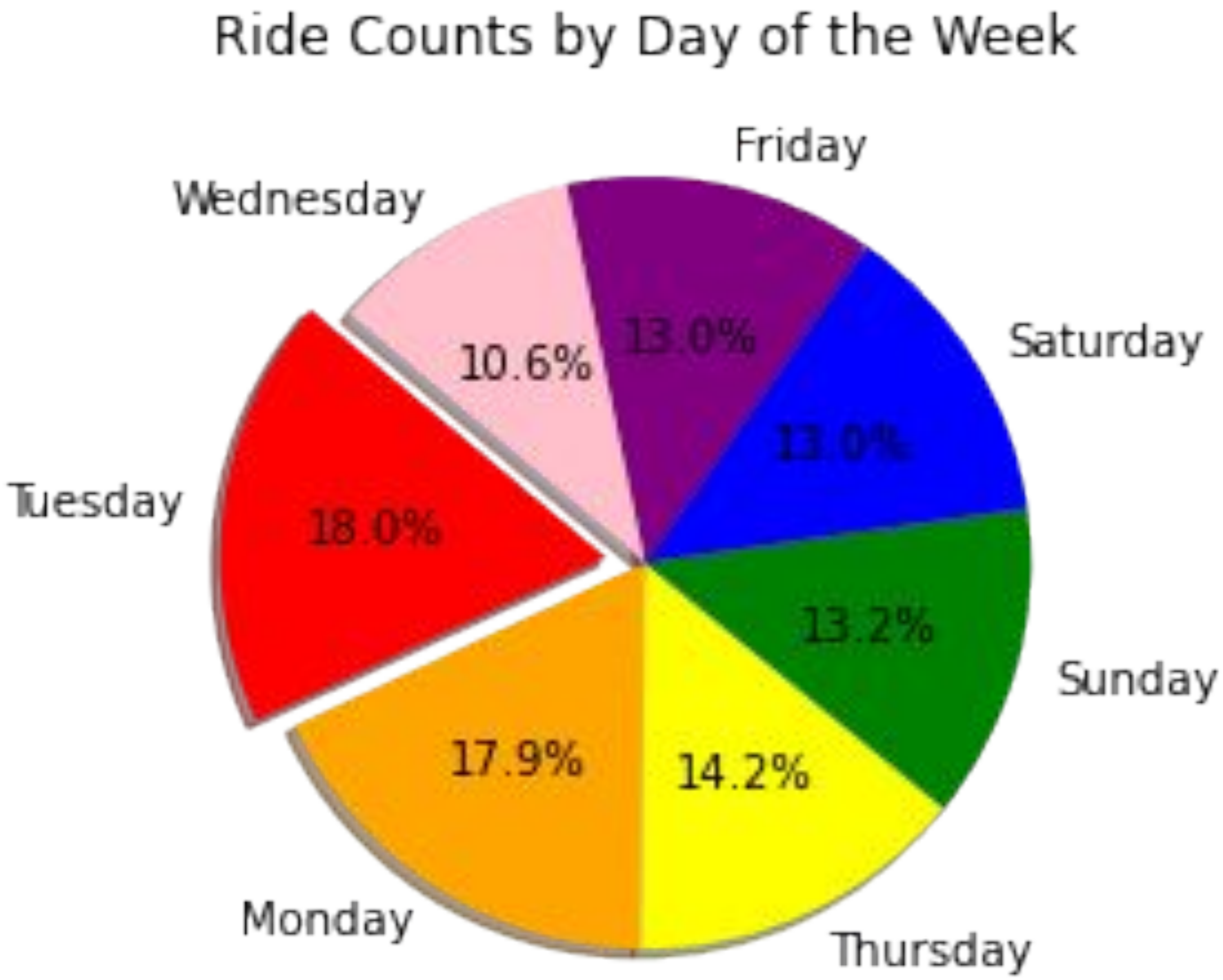
The r-squared is: 0.01682866523852337

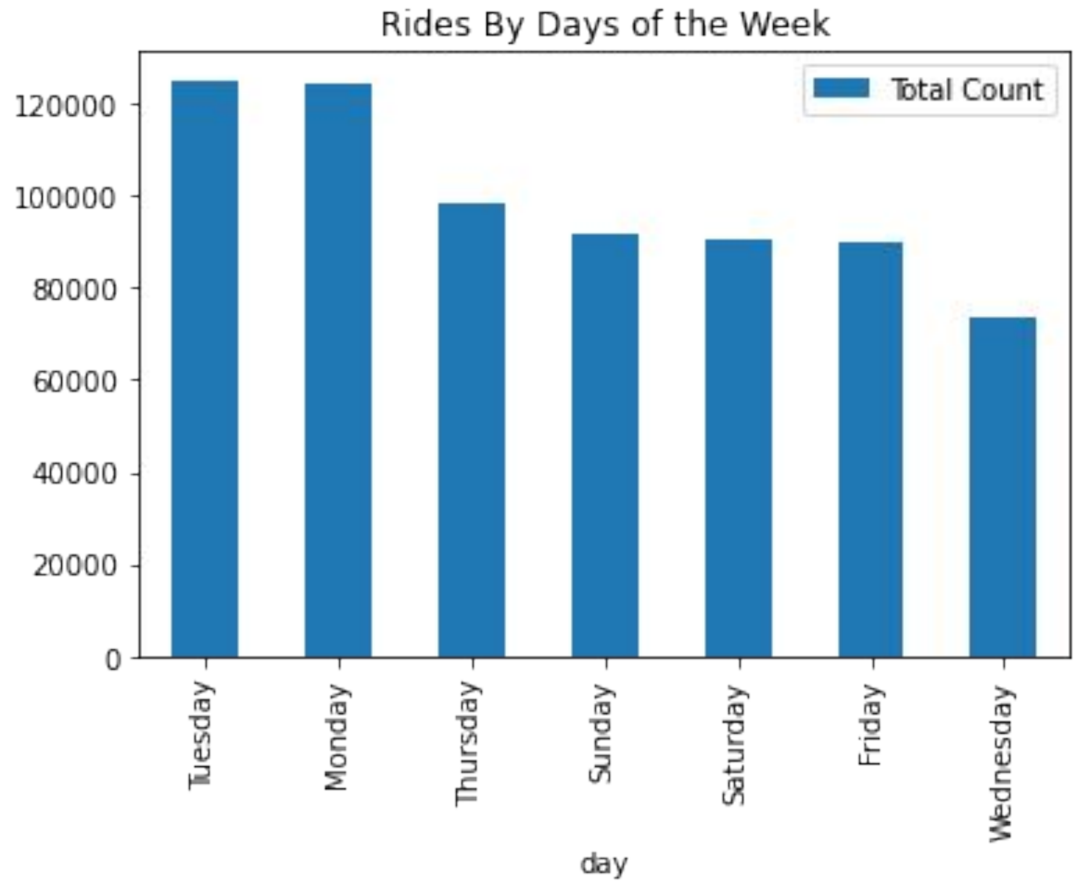



```
The r-squared is: 0.039279321785932825
```


Rides by Time of Day
Total Count
0
20000
40000
60000
80000
Earlier than 2 am
>10 am to noon
>12-2 pm
>4-6 pm
>2-4 pm
>8-10 am
>8-10 pm
>2-4 am
>6-8 pm
>4-6 am
>6-8 am
>10 pm - midnight
Time of Day of Ride


Ride Counts by Time of Day
>6-8 am
>4-6 am
>6-8 pm
>10 pm - midnight
>2-4 am
Earlier than 2 am
>8-10 pm
>10 am to noon
>8-10 am
>12-2 pm
>2-4 pm
>4-6 pm
12.9%
4.6%
7.1%
7.6%
7.8%
8.1%
8.2%
8.6%
8.8%
8.8%
8.8%
8.8%


Ride Counts by Day of the Week
Friday
Wednesday
Saturday
Tuesday
Sunday
Monday
Thursday
10.6%
13.0%
13.0%
18.0%
13.2%
17.9%
14.2%


Rides By Days of the Week
Total Count
120000
100000
80000
60000
40000
20000
0
Tuesday
Monday
Thursday
Sunday
Saturday
Friday
Wednesday
day

Heatmap of Price Points

In Conclusion

We were able to answer our largest questions

What area of Boston are most rides sourced from?

(Geographical Data)

Financial District
8.5% of all rides

What time of day are most rides requested?

(Time Stamp Data)

12:00 - 2:00 AM
12.9% of all rides

What day of the week are rides requested?

(Calendar Data)

Tuesdays
18% of all rides

At what apparent temperature are the most rides requested?

(Weather Data)

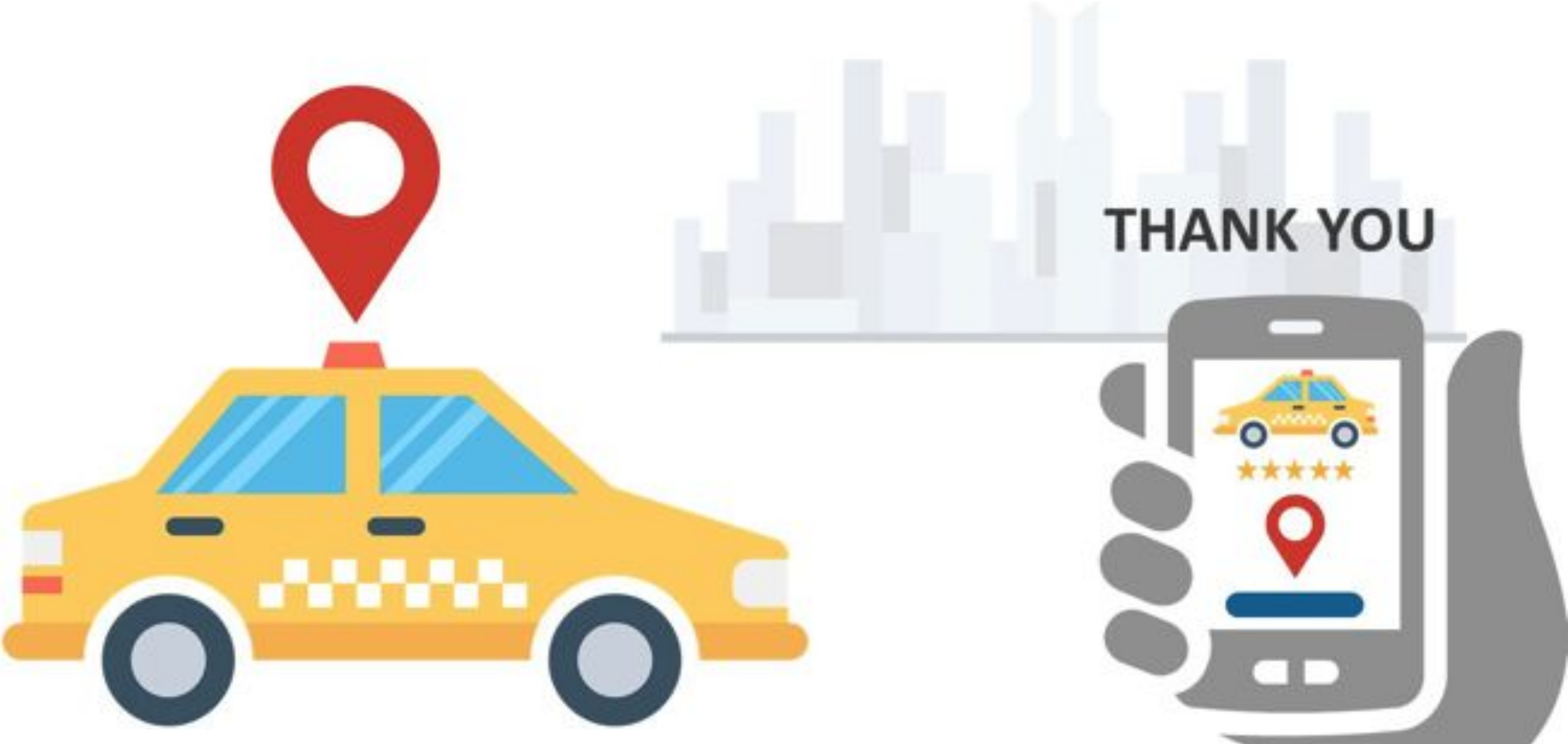
35 (℉) – 40 (℉)
32.2% of all rides

Pricing Data shows that Haymarket Square has the highest prices ($12.39 per mile) and Northeastern University ($7.20 per mile) has the lowest

Post Mortem

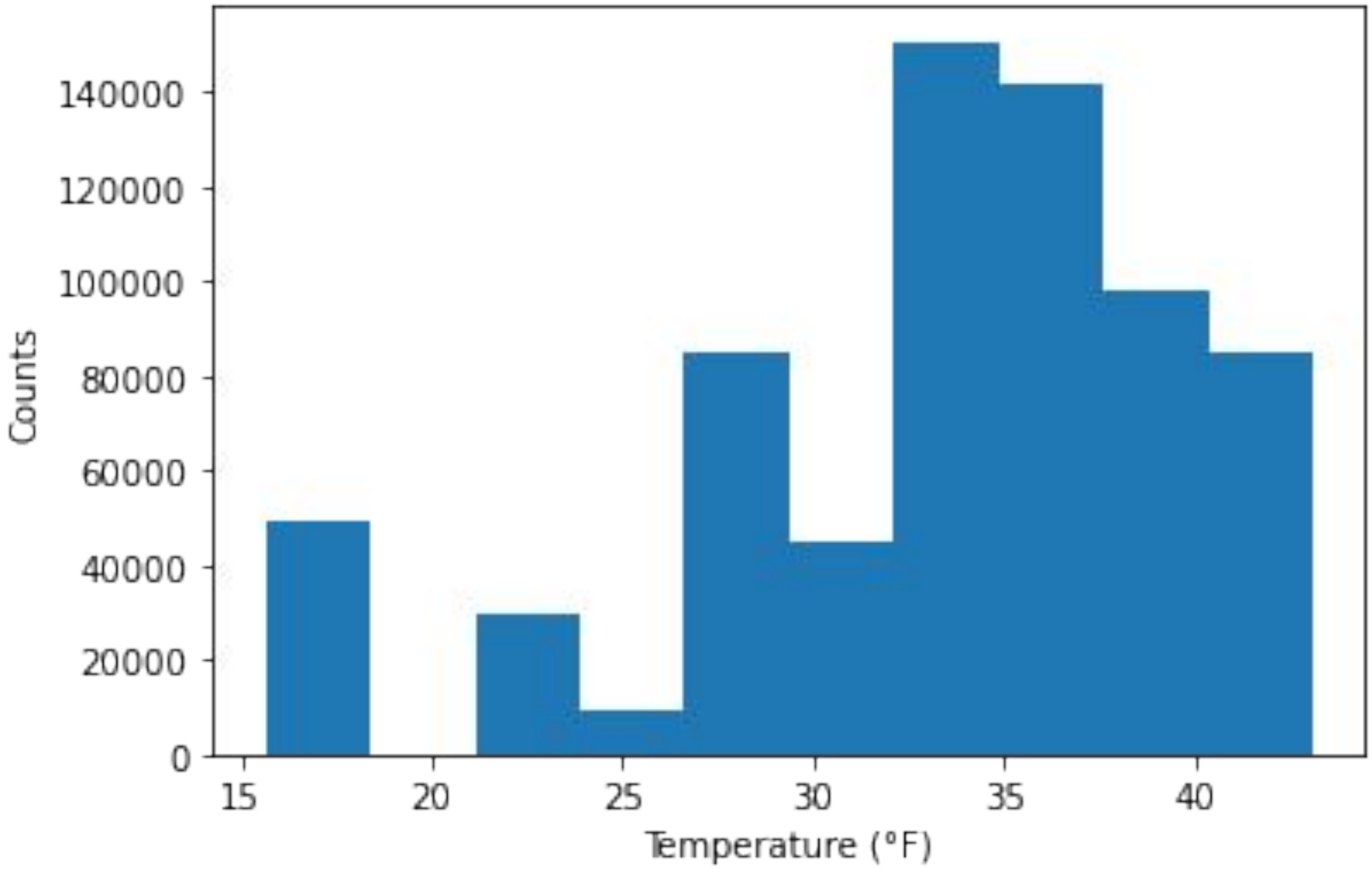
- GitHub Support
    - Large File management (LFS) did not work for all of us and others were disabled/suspended for high use.
- Column definition wasn’t always clear; there was no reference file
- Data was only from 3 weeks, so we assume that these data are an accurate sample of the total population. This is especially concerning/relevant with the weather data. Given the data and the weather data available, we didn’t do a deep dive into the weather patterns.
- Kaggle data sourcing wasn’t clear, but ride ids do seem to represent real data
- We needed to use day of week but we were provided a datetime, so we had to learn the conversion strategy.
- Lacking data for a larger geographical area (for example, Airport)

Questions?



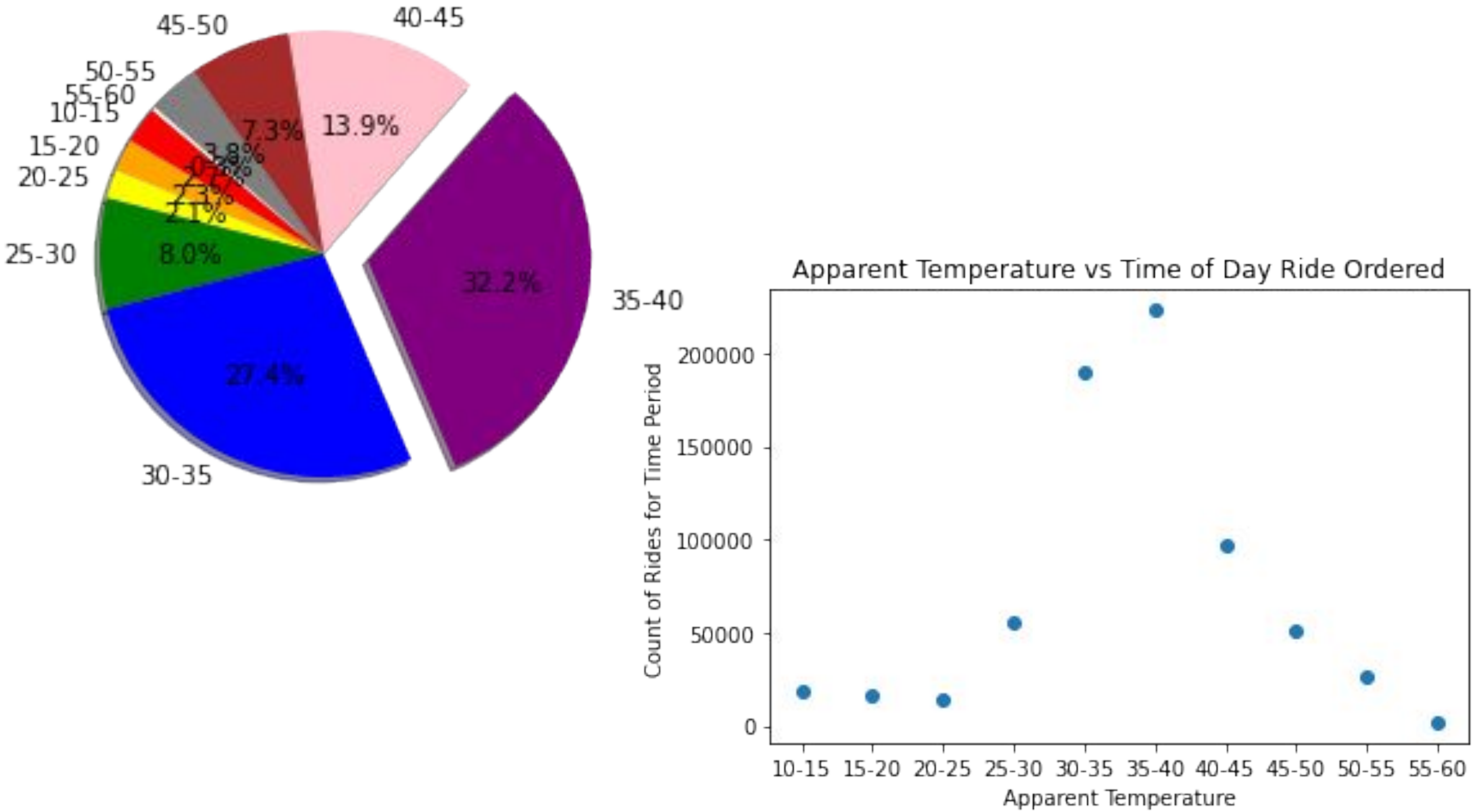
Please see appendix for more supporting visuals

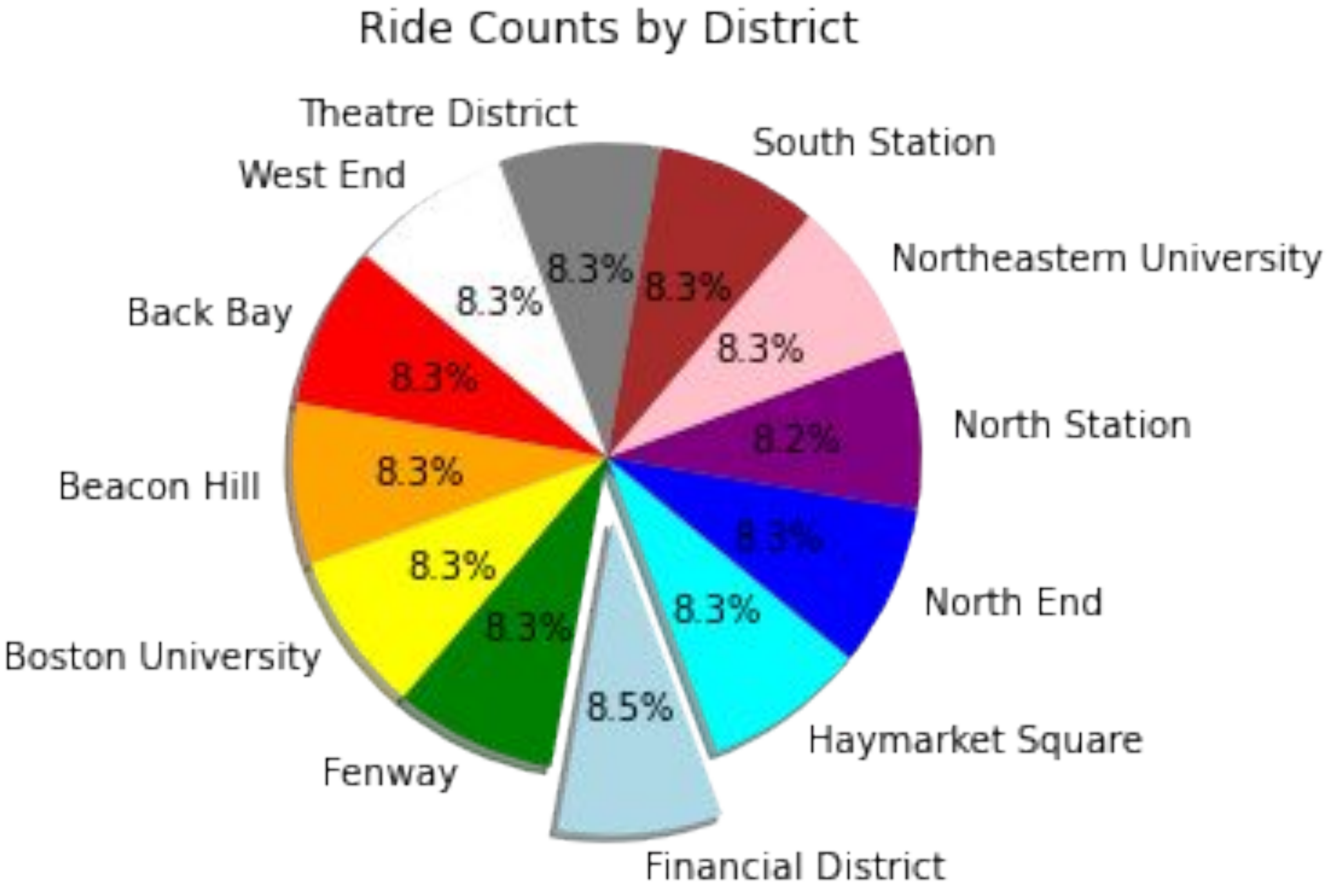
The mean minimum temperature in Boston for the data set is 33.457774355033585
The median minimum temperature in Boston for the data set is 34.24
The mode minimum temperature in Boston for the data set is ModeResult(mode=array([33.7]), count=array([21743]))



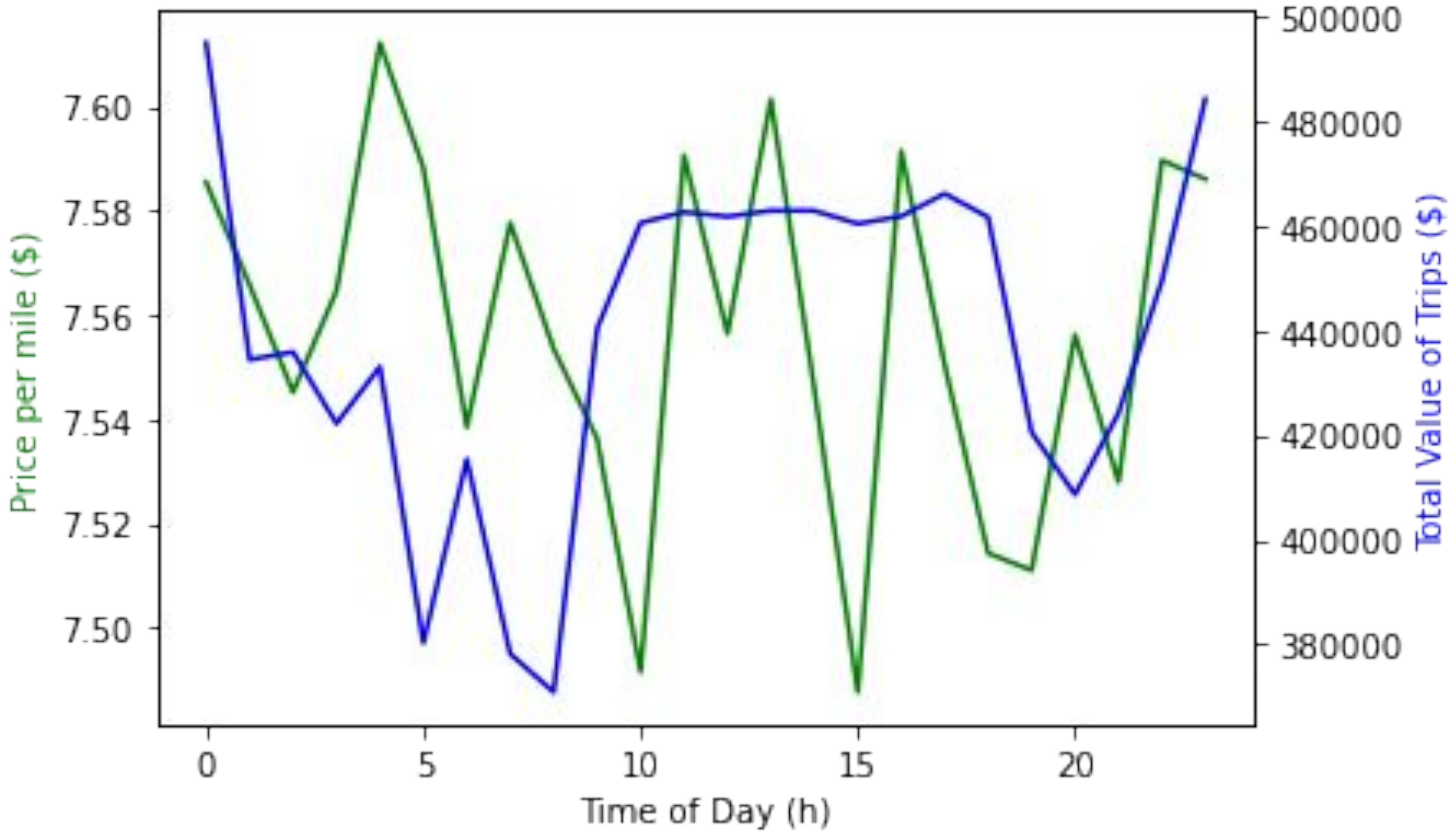
Normaltest Result (statistic=10.791525974654531, pvalue=0.004535758352077577)
The variance using the NumPy module is 41.824926093022235
The standard deviation using the NumPy module is 6.46721934783584
Roughly 68% of the data is between 26.991 and 39.925
Roughly 95% of the data is between 20.523 and 46.392
Roughly 99.7% of the data is between 14.056 and 52.859
The z-scores using the SciPy module are [0.99458907 1.08736464 0.29413347 ... -0.31509282 -0.31509282 -0.31509282]

Resources/Ride Counts by Average Apparent Temperature





	Total Source Counts
source	
Back Bay	57792
Beacon Hill	57403
Boston University	57764
Fenway	57757
Financial District	58857
Haymarket Square	57736
North End	57763
North Station	57118
Northeastern University	57756
South Station	57750
Theatre District	57813
West End	57562



Price per mile ($)
7.60
7.58
7.56
7.54
7.52
7.50
0
5
10
15
20
Time of Day (h)
Total Value of Trips ($)
500000
480000
460000
440000
420000
400000
380000