Cpt_S 315 - NYC Taxi Cab Analysis: Final Report

Libby Stephan, Gurman Grewal, Sabri Tahir, Darrel Nitereka, Joshua Yuan, Logan Mock, Cole Logan, Gabriel Gray

Sabri Tahir - Data Extraction and Sampling
Joshua Yuan - Data Cleaning and Preprocessing
Darrel Nitereka - Summary Statistics
Gabriel Gray - Box Plots and Scatter Plots
Logan Mock - Pearson's Correlation Coefficient
Libby Stephan - ANOVA
Cole Logan - Time Series Analysis
Gurman Grewal - Decision Tree

Introduction

In the bustling streets of New York, taxis are ubiquitous, serving as a highly popular mode of transportation with a rich subculture of their own. For our project, we analyzed a substantial dataset of nearly ten million taxi ride entries. We refined this dataset by removing records with formatting errors and extreme outliers that would not contribute meaningfully to our analysis. The aim of this project was to explore how seasonal changes and the rise of rideshare companies like Uber and Lyft have influenced taxi usage. We also examined the impact of the COVID-19 pandemic on taxi services. Our goal was to derive insights that could help enhance and grow taxi-based services in the city.

Motivation

Our team thought that the Taxi record dataset would be an interesting and complex project that would offer lots of insight on a real world service while also providing excellent experience in using what we learned from class.

Questions to Investigate

- How seasonalities impact tip amount, trip distance
- How the number of passengers impact tips, trip distance
- Whether the trip distance influences the tip amount
- How Covid impacted tip amount, trip distance, etc
- How Uber impacted tip amount, trip distance, etc

Challenges

The dataset provided was extremely large and contained a lot of noise. This called for a lot of data processing and cleaning. This was the main challenge we faced in the duration of this project.

- **Summary statistics**
- **Time Series analysis**
  - This section was quite straightforward. The only issues that came up were simple troubleshooting matters with the matplot library.
- **ANOVA**
  - Fulfilling the assumptions of the ANOVA proved to be difficult due to the non-normal distribution of the data. This was mitigated through the use of bootstrap sampling.
- **Decision tree**

- **Scatter plots/box plots**
  - For this the major issue was trying to make the graphs look good and understandable while still keeping the integrity of the data so making sure to cut out the outliers in the data so we could make a functional best fit line for each graph.
- **Pearson's Correlation Coefficient**
  - The cleaning and the preprocessing of the data was very sufficient which resulted in no challenges in computing the correlation coefficients.

Summarization of Results

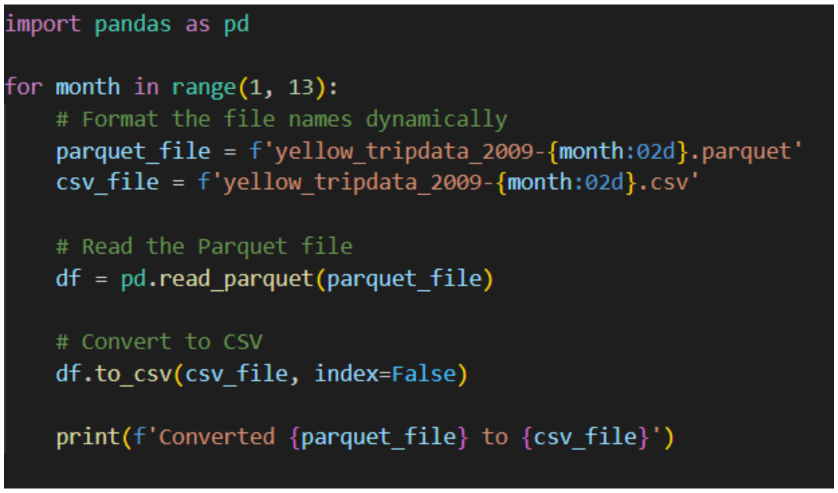
Our project successfully explored the impact of seasonal change, the growth of rideshare companies (Uber, Lyft), and the effects of the pandemic in 2020 on the New York taxi industry. Based on a smaller sample we found that during the pandemic there were lower trip distances and tip amounts which often are linearly connected, but looking at the whole population there was no difference.

- Seasons do not appear to truly impact tip percentage, but it may impact trip distance such that people are likely to travel longer distances in the summer than winter.
- Single passengers are likely to tip higher while traveling less distance as opposed to multiple passengers leaning towards lower tip percentage and higher travel distances.
- Covid did not appear to impact tip percentage after 2019, however, tipping did go up since 2020. In terms of trip distance, the average distance is lower in 2020 than in 2019 and 2023.
- There is no difference in tips or trip distance between 2009 and 2011 implying that the introduction of Uber did not immediately alter anything
- Season x Taxi Frequency correlation: -.8233
- Season x Tip Amount correlation: 0.8782
- Season x Taxi Distance correlation: 0.3524
- # of Passengers x Tips correlation: -0.0323
- # of Passengers x Distance correlation: -0.0003
- Tip Amount x Distances correlation: 0.0396
- Tip Amount x Fair Amount correlation: 0.5460

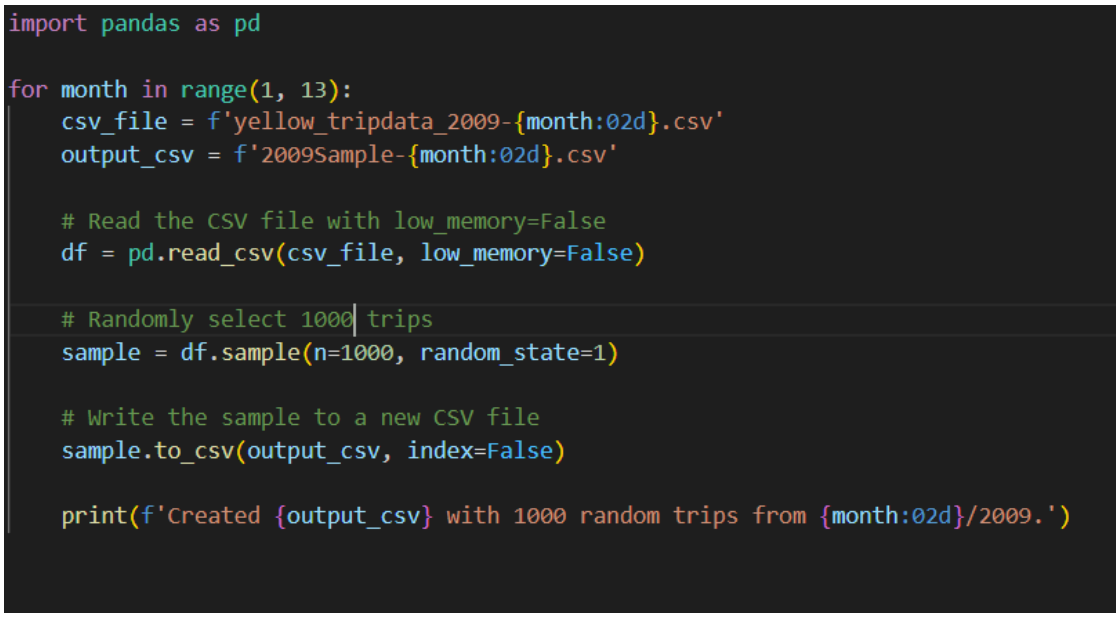
Data Mining Task

<u>DATA EXTRACTION - PROCESS PARQUET FILES TO CSV</u>

The files we retrieved from the NYC Yellow Cab taxi page were in .parquet, and we wanted them in .csv so we could perform analysis on the data. Below are the 2 code snippets that a) converted the files to csv and b) transformed the csv files into usable, smaller sample files. This was repeated for each year of information we wanted to retrieve, and we were left with 6 folders filled with 12 months of 1000 entries each.



```python
import pandas as pd

for month in range(1, 13):
    # Format the file names dynamically
    parquet_file = f'yellow_tripdata_2009-{month:02d}.parquet'
    csv_file = f'yellow_tripdata_2009-{month:02d}.csv'

    # Read the Parquet file
    df = pd.read_parquet(parquet_file)

    # Convert to CSV
    df.to_csv(csv_file, index=False)

    print(f'Converted {parquet_file} to {csv_file}')
```



```python
import pandas as pd

for month in range(1, 13):
    csv_file = f'yellow_tripdata_2009-{month:02d}.csv'
    output_csv = f'2009Sample-{month:02d}.csv'

    # Read the CSV file with low_memory=False
    df = pd.read_csv(csv_file, low_memory=False)

    # Randomly select 1000 trips
    sample = df.sample(n=1000, random_state=1)

    # Write the sample to a new CSV file
    sample.to_csv(output_csv, index=False)

    print(f'Created {output_csv} with 1000 random trips from {month:02d}/2009.')
```

DATA PREPROCESSING & CLEANING

Standardization

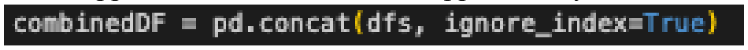
Upon an initial scan of the .csv files we now had, we observed discrepancies in the terminology of column names across datasets spanning from 2009 - 2023. For example, the 2009 dataset labeled pickup times as ‘Trip_Pickup_DateTime’ whereas datasets from 2011 onwards used ‘tpep_pickup_datetime’. To join our datasets for combined analysis we first standardized the column names in the 2009 dataset to match those from 2011 to 2023.

This was done by identifying equivalent columns across all datasets and renaming the columns in the 2009 dataset to align with the labels of the later datasets.

For this use case, standardization was critical to ensure consistency in our next step of data integration

Integration

All individual datasets were consolidated into a unified dataset using Python Pandas library. Using the code snippet below, we were able to append every dataset into a singular dataframe.

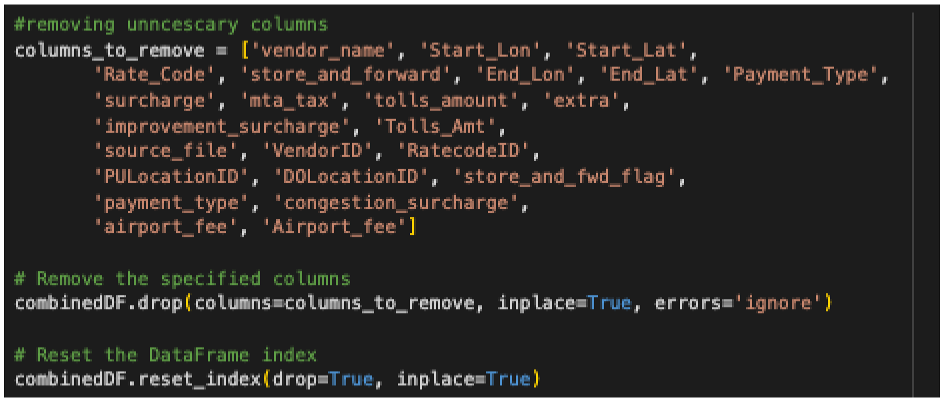


```
combinedDF = pd.concat(dfs, ignore_index=True)
```

Data Cleaning

Removing Irrelevant Columns

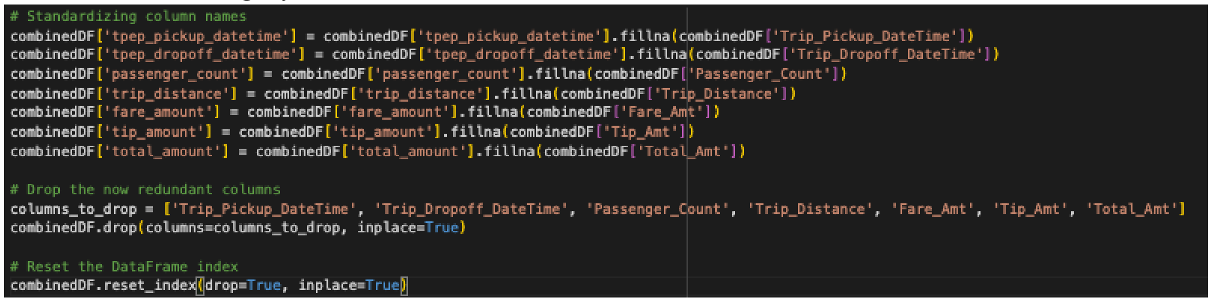
Several columns that were not required for analysis, such as ‘vendor_name’, ‘PULocationID’, and ‘DOLocationID’ were removed. This step reduced the complexity and size of the dataset, focusing on relevant data only.



```
#removing unncescary columns
columns_to_remove = ['vendor_name', 'Start_Lon', 'Start_Lat',
       'Rate_Code', 'store_and_forward', 'End_Lon', 'End_Lat', 'Payment_Type',
       'surcharge', 'mta_tax', 'tolls_amount', 'extra',
       'improvement_surcharge', 'Tolls_Amt',
       'source_file', 'VendorID', 'RatecodeID',
       'PULocationID', 'DOLocationID', 'store_and_fwd_flag',
       'payment_type', 'congestion_surcharge',
       'airport_fee', 'Airport_fee']

# Remove the specified columns
combinedDF.drop(columns=columns_to_remove, inplace=True, errors='ignore')

# Reset the DataFrame index
combinedDF.reset_index(drop=True, inplace=True)
```

Correcting Data Types and Handling Null Values

The data types for date and numeric columns were standardized. Null values in crucial columns like tpep_pickup_datetime and passenger_count were identified and rows with these null values were removed to maintain data integrity.



```
# Standardizing column names
combinedDF['tpep_pickup_datetime'] = combinedDF['tpep_pickup_datetime'].fillna(combinedDF['Trip_Pickup_DateTime'])
combinedDF['tpep_dropoff_datetime'] = combinedDF['tpep_dropoff_datetime'].fillna(combinedDF['Trip_Dropoff_DateTime'])
combinedDF['passenger_count'] = combinedDF['passenger_count'].fillna(combinedDF['Passenger_Count'])
combinedDF['trip_distance'] = combinedDF['trip_distance'].fillna(combinedDF['Trip_Distance'])
combinedDF['fare_amount'] = combinedDF['fare_amount'].fillna(combinedDF['Fare_Amt'])
combinedDF['tip_amount'] = combinedDF['tip_amount'].fillna(combinedDF['Tip_Amt'])
combinedDF['total_amount'] = combinedDF['total_amount'].fillna(combinedDF['Total_Amt'])

# Drop the now redundant columns
columns_to_drop = ['Trip_Pickup_DateTime', 'Trip_Dropoff_DateTime', 'Passenger_Count', 'Trip_Distance', 'Fare_Amt', 'Tip_Amt', 'Total_Amt']
combinedDF.drop(columns=columns_to_drop, inplace=True)

# Reset the DataFrame index
combinedDF.reset_index(drop=True, inplace=True)
```

Handling Irregular Values

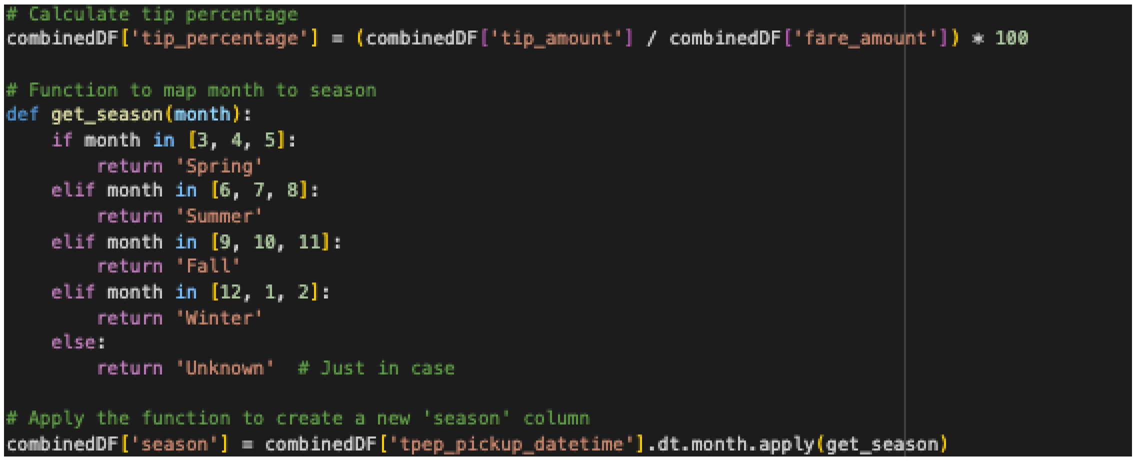
Entries that had nonsensical values such as a negative number of passengers or a fee greater than 100% were removed from the data set.

Feature Engineering

To dig deeper into what the data really tells us, we engineered two new features that we think would yield useful information for further analysis:

Tip Percentage: Calculated as the ratio of tip to the total fare expressed as a percentage value.

Seasonality: Derived from ‘tpep_pickup_datetime’ by mapping the month of the pickup to a corresponding season.


```
# Calculate tip percentage
combinedDF['tip_percentage'] = (combinedDF['tip_amount'] / combinedDF['fare_amount']) * 100

# Function to map month to season
def get_season(month):
    if month in [3, 4, 5]:
        return 'Spring'
    elif month in [6, 7, 8]:
        return 'Summer'
    elif month in [9, 10, 11]:
        return 'Fall'
    elif month in [12, 1, 2]:
        return 'Winter'
    else:
        return 'Unknown'  # Just in case

# Apply the function to create a new 'season' column
combinedDF['season'] = combinedDF['tpep_pickup_datetime'].dt.month.apply(get_season)
```

SUMMARY STATISTICS

Objective

The objective in analyzing summary statistics was to establish a foundational understanding of the NYC taxi trip data, focusing on key metrics such as trip distances, payment types, and rider counts. These metrics provide insights into the central tendencies and variability of the data, crucial for subsequent predictive modeling and decision-making processes.

Challenges

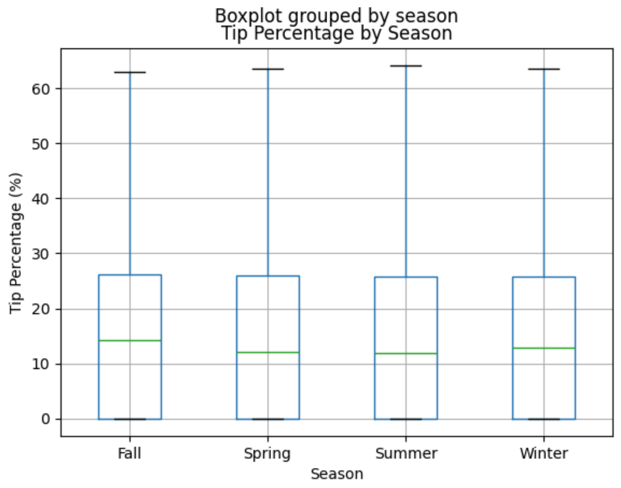
A significant challenge was dealing with outliers. These outliers, particularly in trip distances, posed questions regarding data entry errors versus genuine long-distance trips. Their presence skewed the data distribution, complicating the interpretation of average values.

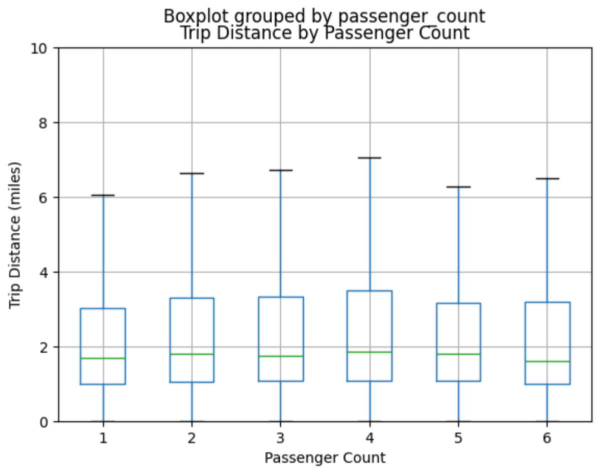
Approach

I employed various statistical techniques to summarize the data. These included calculating measures of central tendency (mean, median) and dispersion (standard deviation, interquartile range), and employing visualizations like histograms and box plots to better understand the distribution of trip distances.

BOX PLOTS AND SCATTER PLOTS

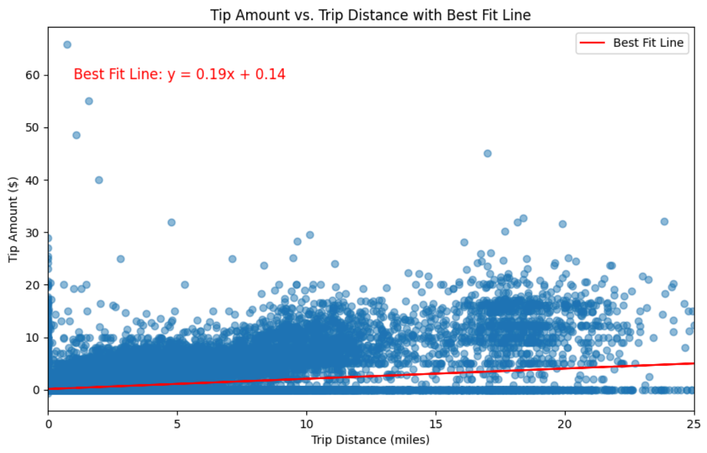
In our analysis, we utilized box plots and scatter plots to effectively examine the distribution of key variables within our dataset. Our results showed that while the highest individual tips were noted during the summer months, the fall season yielded the highest average tips overall. We theorize that this trend could be influenced by seasonal mood variations, where the pleasant weather of summer and fall may increase people's willingness to tip generously. Furthermore, our findings indicated that trips involving groups of four either recorded the shortest or the longest distances, suggesting that larger groups are more inclined to undertake longer journeys, which contribute to the higher average and maximum distances seen in our dataset. Additionally, while our box plot analysis highlighted differences in data distribution across seasons, our ANOVA focused on discerning population differences, providing a complementary perspective by examining variations within sample groups.

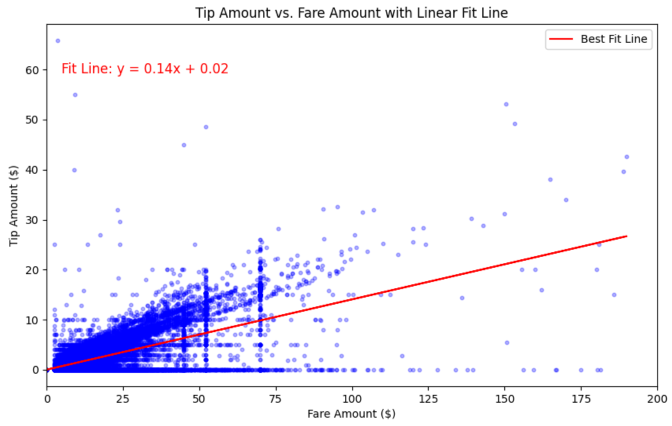




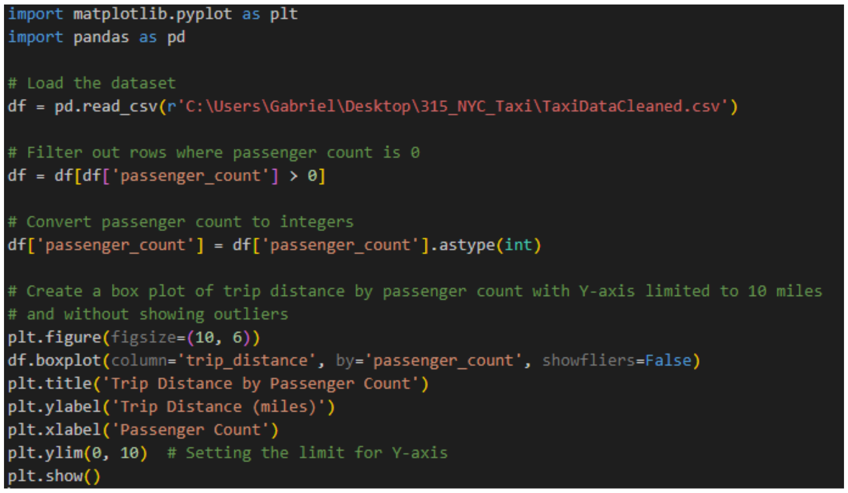
Our analysis utilized scatter plots to examine the relationships between tip amount and trip distance, as well as tip amount and fare amount. These plots revealed numerous outliers, with instances of both unusually low tips for long distances and unusually high tips for short trips. Notably, nearly 24% of passengers did not tip at all, regardless of the distance traveled. This observation was consistent across both datasets. The overall trends from the scatter plots suggested a linear relationship between the variables, indicating that tip amounts generally increase with both fare amount and trip distance. To refine our analysis and better understand these relationships, we created best-fit lines for the data, deliberately excluding the 24% of non-tipping passengers to avoid skewing the results. This approach helped clarify how tipping behaviors vary with changes in fare and distance under typical circumstances.

Here is the code we used for generating the plots and for computing the best-fit lines, which specifically exclude outliers for more accurate analysis.



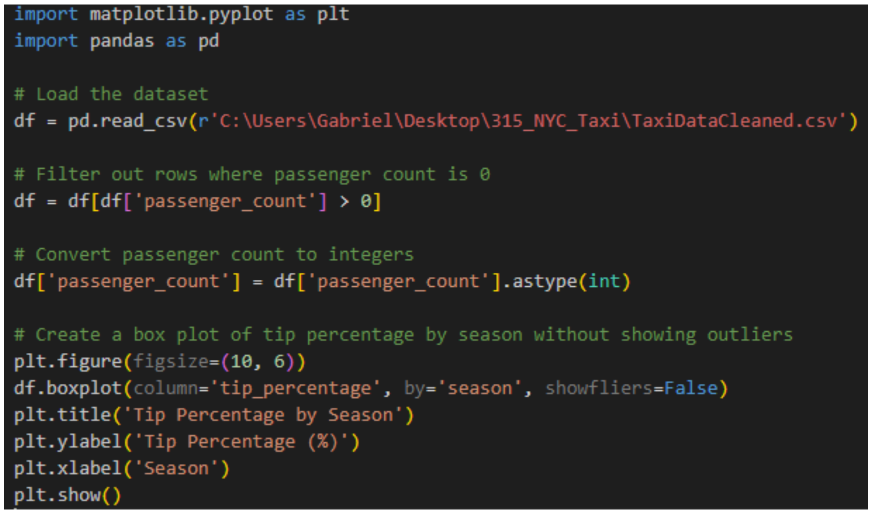


BoxPlot Trip Distance by Passenger Count



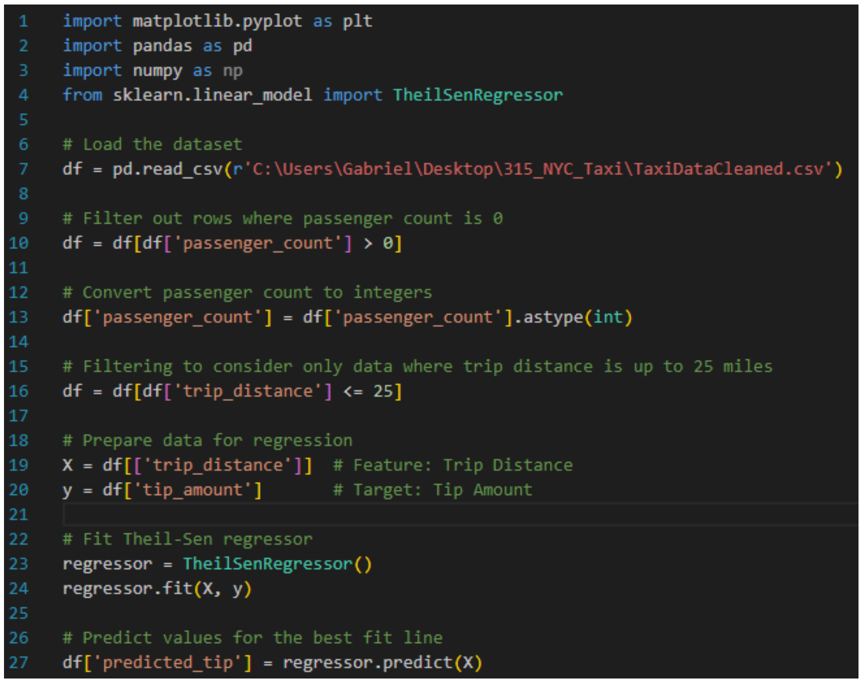
```python
import matplotlib.pyplot as plt
import pandas as pd

# Load the dataset
df = pd.read_csv(r'C:\Users\Gabriel\Desktop\315_NYC_Taxi\TaxiDataCleaned.csv')

# Filter out rows where passenger count is 0
df = df[df['passenger_count'] > 0]

# Convert passenger count to integers
df['passenger_count'] = df['passenger_count'].astype(int)

# Create a box plot of trip distance by passenger count with Y-axis limited to 10 miles
# and without showing outliers
plt.figure(figsize=(10, 6))
df.boxplot(column='trip_distance', by='passenger_count', showfliers=False)
plt.title('Trip Distance by Passenger Count')
plt.ylabel('Trip Distance (miles)')
plt.xlabel('Passenger Count')
plt.ylim(0, 10)  # Setting the limit for Y-axis
plt.show()
```

BoxPlot Tip Percentage by Season

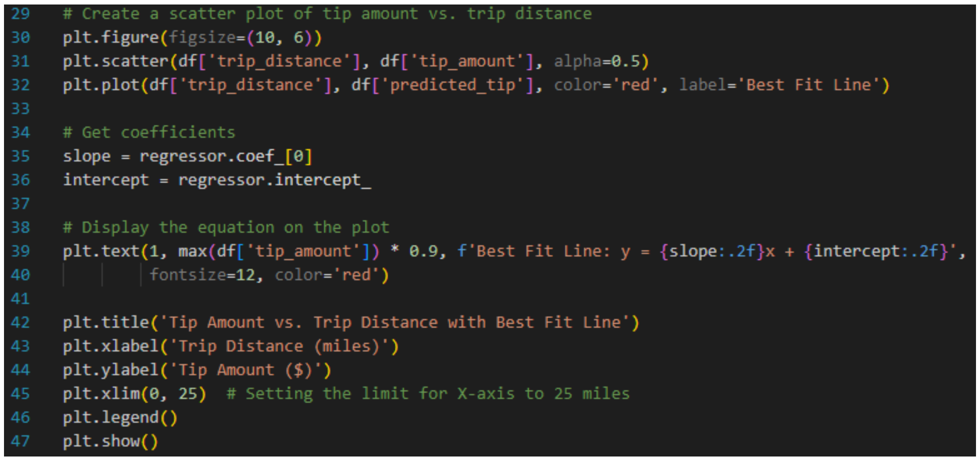


```python
import matplotlib.pyplot as plt
import pandas as pd

# Load the dataset
df = pd.read_csv(r'C:\Users\Gabriel\Desktop\315_NYC_Taxi\TaxiDataCleaned.csv')

# Filter out rows where passenger count is 0
df = df[df['passenger_count'] > 0]

# Convert passenger count to integers
df['passenger_count'] = df['passenger_count'].astype(int)

# Create a box plot of tip percentage by season without showing outliers
plt.figure(figsize=(10, 6))
df.boxplot(column='tip_percentage', by='season', showfliers=False)
plt.title('Tip Percentage by Season')
plt.ylabel('Tip Percentage (%)')
plt.xlabel('Season')
plt.show()
```

ScatterPlot Tip Amount by Trip Distance with Best Fit Line

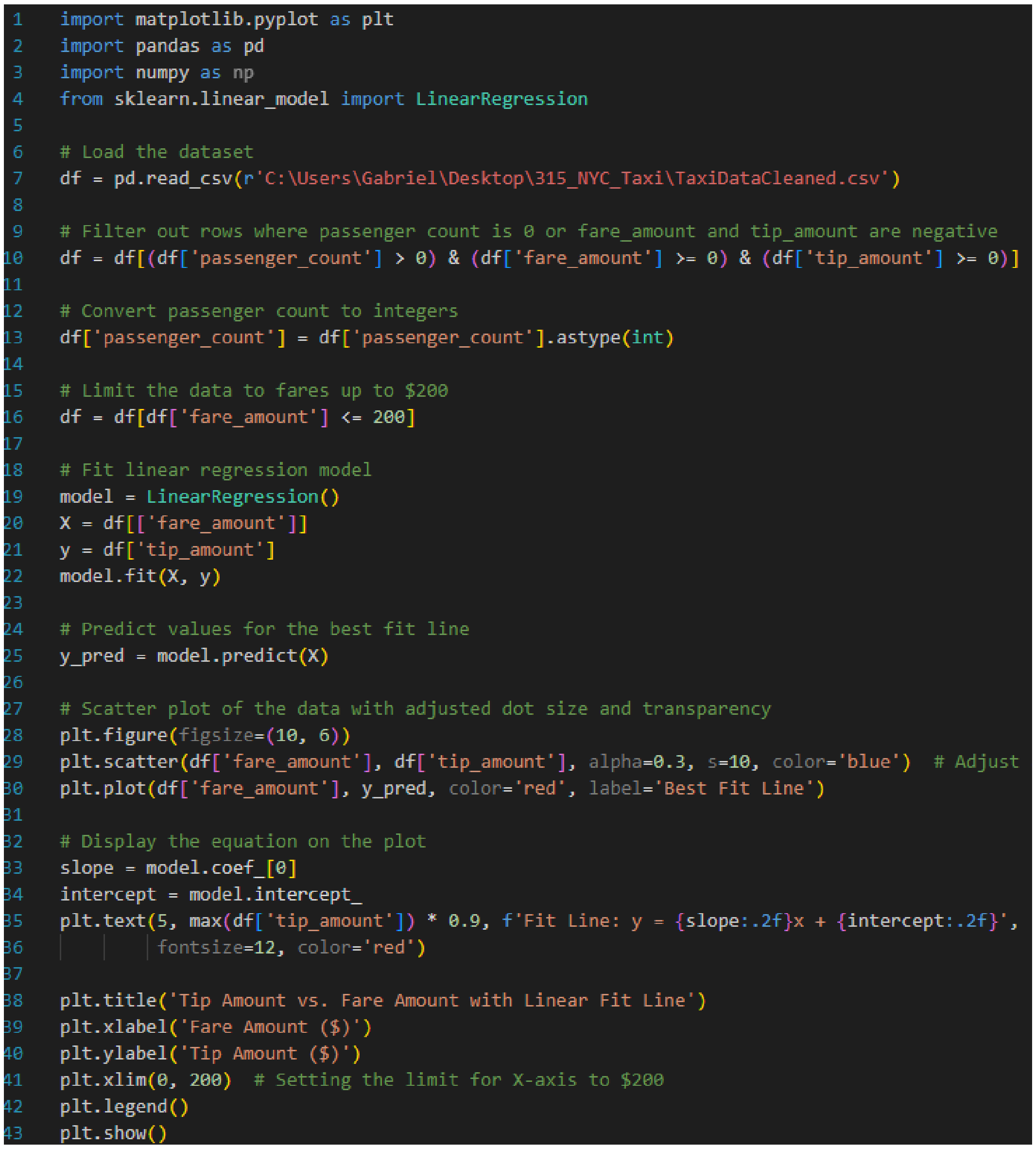


```python
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.linear_model import TheilSenRegressor

# Load the dataset
df = pd.read_csv(r'C:\Users\Gabriel\Desktop\315_NYC_Taxi\TaxiDataCleaned.csv')

# Filter out rows where passenger count is 0
df = df[df['passenger_count'] > 0]

# Convert passenger count to integers
df['passenger_count'] = df['passenger_count'].astype(int)

# Filtering to consider only data where trip distance is up to 25 miles
df = df[df['trip_distance'] <= 25]

# Prepare data for regression
X = df[['trip_distance']]  # Feature: Trip Distance
y = df['tip_amount']       # Target: Tip Amount

# Fit Theil-Sen regressor
regressor = TheilSenRegressor()
regressor.fit(X, y)

# Predict values for the best fit line
df['predicted_tip'] = regressor.predict(X)
```



```python
# Create a scatter plot of tip amount vs. trip distance
plt.figure(figsize=(10, 6))
plt.scatter(df['trip_distance'], df['tip_amount'], alpha=0.5)
plt.plot(df['trip_distance'], df['predicted_tip'], color='red', label='Best Fit Line')

# Get coefficients
slope = regressor.coef_[0]
intercept = regressor.intercept_

# Display the equation on the plot
plt.text(1, max(df['tip_amount']) * 0.9, f'Best Fit Line: y = {slope:.2f}x + {intercept:.2f}',
         fontsize=12, color='red')

plt.title('Tip Amount vs. Trip Distance with Best Fit Line')
plt.xlabel('Trip Distance (miles)')
plt.ylabel('Tip Amount ($)')
plt.xlim(0, 25)  # Setting the limit for X-axis to 25 miles
plt.legend()
plt.show()
```

ScatterPlot Tip Amount by Fare Price



```python
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression

# Load the dataset
df = pd.read_csv(r'C:\Users\Gabriel\Desktop\315_NYC_Taxi\TaxiDataCleaned.csv')

# Filter out rows where passenger count is 0 or fare_amount and tip_amount are negative
df = df[(df['passenger_count'] > 0) & (df['fare_amount'] >= 0) & (df['tip_amount'] >= 0)]

# Convert passenger count to integers
df['passenger_count'] = df['passenger_count'].astype(int)

# Limit the data to fares up to $200
df = df[df['fare_amount'] <= 200]

# Fit linear regression model
model = LinearRegression()
X = df[['fare_amount']]
y = df['tip_amount']
model.fit(X, y)

# Predict values for the best fit line
y_pred = model.predict(X)

# Scatter plot of the data with adjusted dot size and transparency
plt.figure(figsize=(10, 6))
plt.scatter(df['fare_amount'], df['tip_amount'], alpha=0.3, s=10, color='blue')  # Adjust
plt.plot(df['fare_amount'], y_pred, color='red', label='Best Fit Line')

# Display the equation on the plot
slope = model.coef_[0]
intercept = model.intercept_
plt.text(5, max(df['tip_amount']) * 0.9, f'Fit Line: y = {slope:.2f}x + {intercept:.2f}',
         fontsize=12, color='red')

plt.title('Tip Amount vs. Fare Amount with Linear Fit Line')
plt.xlabel('Fare Amount ($)')
plt.ylabel('Tip Amount ($)')
plt.xlim(0, 200)  # Setting the limit for X-axis to $200
plt.legend()
plt.show()
```

TIME SERIES ANALYSIS

Time series analysis aims to view data graphed in order of time. These graphs allow us to glean information about how certain attributes of our taxi data change over time.

For our purposes, we chose to view each of our taxi data attributes over the 12 months of the year, the 4 seasons, and the 15 years covered by our data. The best way to graph our data in this situation was to take the average of each attribute over the period of time and then plot those averages.

Some questions that we set out to answer with these graphs were:

- Do the number of taxi rides decrease as we see rideshare apps come into existence?
- Does change in the weather (season) affect the number of passengers in rides?
- Do we see the fare of taxi rides increasing with inflation as years pass?
- Do certain months/holidays cause any peculiar trends?

There were not too many challenges taking on this task, other than the fact that it was tedious. This task consisted of using trial and error to get the first graph configured correctly with the library we chose to use (Matplotlib), then doing the same thing 17 more times.

ANOVA

ANOVA, which stands for Analysis of Variance, determines whether differences between sample group means are statistically significant or not. In other words, it parses whether these differences accurately reflect the population at large, or whether it was merely these particular instances sampled to obtain these results. It is a type of hypothesis testing where the null hypothesis, H_0, states that all group population means are the same (e.g. $\mu_1 = \mu_2 = \mu_3$). The alternative hypothesis, H_A, opposes the null hypothesis with the claim that at least one pair of population means differs from each other (e.g. $\mu_1 \neq \mu_2 \vee \mu_1 \neq \mu_3 \vee \mu_2 \neq \mu_3$). The null hypothesis is rejected if the p-value, or the probability of obtaining these results or more extreme results, is less than some $\boldsymbol{\alpha}$ value. $\boldsymbol{\alpha}$ is a chosen value, but it most frequently is picked to be 0.05, so 0.05 will be the $\boldsymbol{\alpha}$ used for this approach.

ANOVA requires three assumptions to be made in order for its results to be considered reliable. The first states that each group follows a normal distribution. The second requires that each group has roughly equal variances. Lastly, all samples must be independent of each other.

It is important to note that if an ANOVA test rejects the null hypothesis, it does not show which pair of means is truly different, simply that a difference exists among the groups. To determine the pair that is different, post hoc testing in the form of a Tukey test is used. Tukey tests will iterate through all pairs of the groups and list which among them are statistically significant.

The questions that will be answered using ANOVA are:

- Do seasons impact tip percentage or trip distance?
- Do the number of passengers impact tip percentage or trip distance?
- Did Covid impact tip percentage or trip distance?
- Did Uber impact tip percentage or trip distance?

The biggest challenge of using ANOVA stems from the inability to satisfy all three assumptions. In particular, having an approximately normal distribution in all groups is not likely when dealing with financial data such as tip percentage as there is simply a select few people who will tip much higher than the rest. This skew can be easily magnified in a city like New York where the wealth gap is more prevalent.

Technical Approach

<u>TIME SERIES ANALYSIS</u>

To carry out our time series analysis the Matplotlib library was used, more specifically the plt attribute.

```python
import matplotlib.pyplot as plt
```

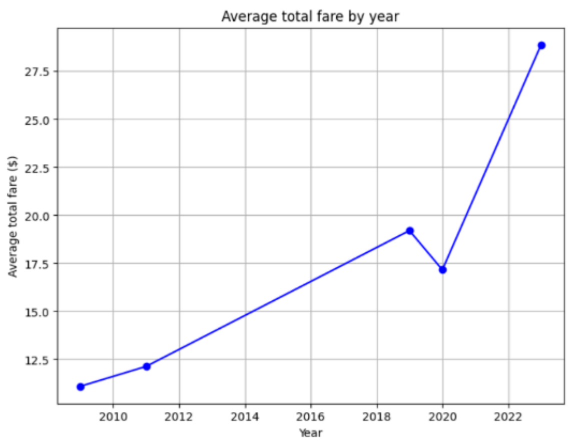
Figure 0.0 - Import statement for plotting

The graphs were simple line graphs and uniform in size and format. Figure 0.1 shows the code used to plot one of the 18 graphs created.

```python
# average trip distance by season
plt.figure(figsize=(8, 6))
avg_distance = combinedDF.groupby('season', sort=False)['trip_distance'].mean()
plt.plot(seasons, avg_distance, marker='o', linestyle='-', color='b')
plt.xlabel('Season')
plt.ylabel('Average trip distance (miles)')
plt.title('Average trip distance by season')
plt.grid(True)
plt.show()
```

Figure 0.1 - Code that creates, labels, and displays a graph

Of the 18 graphs we acquired, only a few really gave us any valuable information. Below are 4 of the most interesting and useful.



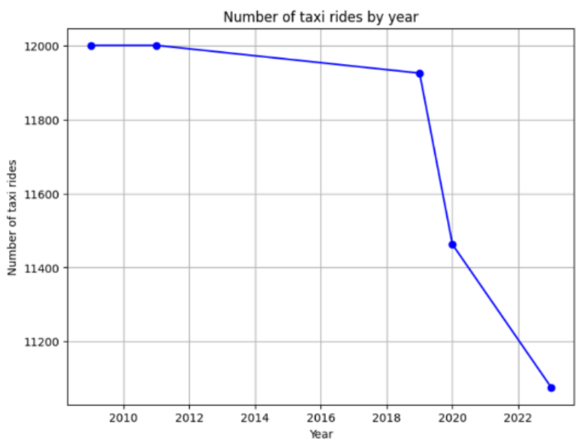
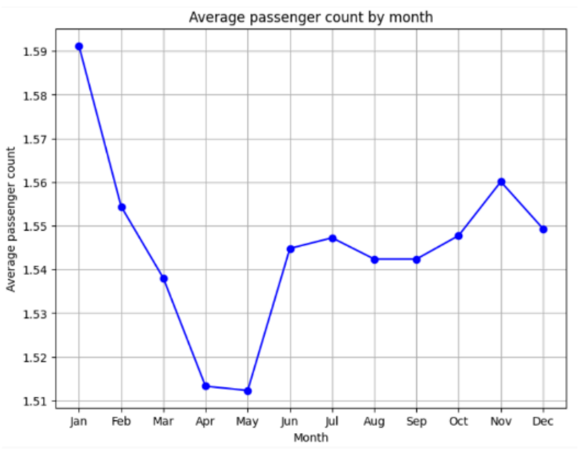


Figure 0.2 (left) - Average total fare by year & Figure 0.3 (right) - Number of taxi rides by year



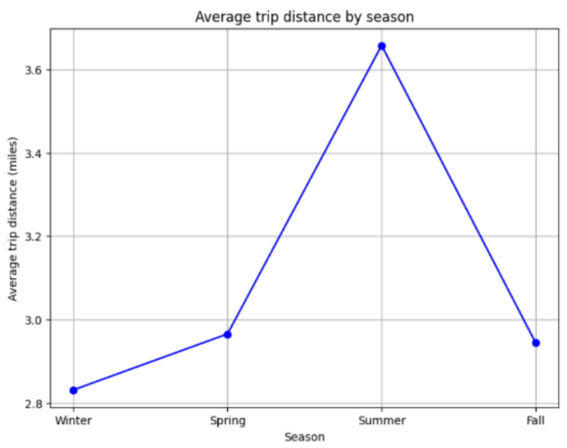


Figure 0.4 (left) - Average passenger #'s by month & Figure 0.5 (right) - Average trip distance by season

Figure 0.2 displays the increase in total fare amount over the years we had data for. This answers one of the questions we set out to answer, whether or not the fare increases with inflation. Surprisingly, the fare increases at a higher rate than inflation. We would expect to see 2023's averages hover around $17 if it closely followed inflation, butas we see it is over $10 over that prediction. It is important to note that total fare includes the fare for the ride plus the tip amount.

Figure 0.3 shows the number of taxi rides decreasing over the years. However, you can see that the y axis scale makes the dropoff look much more staggering than it is in reality. But, the dropoff is still there, and we can see that the average number of taxi rides have decreased.

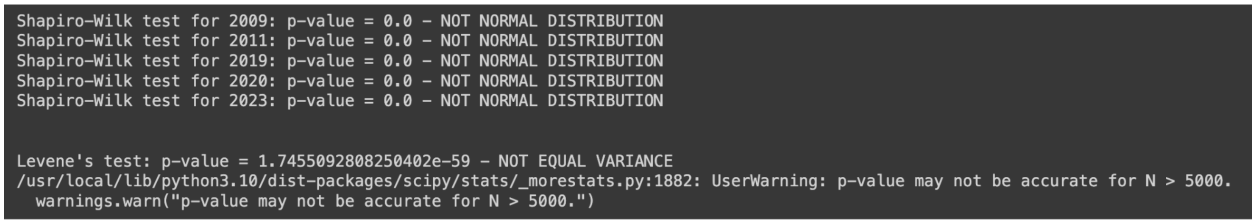
Figure 0.4 follows the average number of passengers over each month in the year. In January there seems to be a higher number of people sharing rides with others, possibly due to the cold or the holiday season. However, just as is the case with the previous graph, the y axis scale makes the difference seem much larger than it is.

Figure 0.5 shows trip distance by season. We can see that New Yorkers tend to take longer taxi rides in the summertime and shorter taxi rides in the winter. It is very interesting to see that the difference between summer and winter is close to a full mile.

ANOVA

From the three assumptions, independent samples is the only quality that can be easily assumed as the behavior of one taxi ride will not affect the behavior of another taxi ride. However, there are two tests to verify normal distributions and equal variances among the groups: Shapiro-Wilk test and Levene's test for normality and equal variance respectively. However, there is another version of the standard ANOVA that does not require equal variances. This version is Welch's ANOVA which will be used if all but the equal variance assumption are satisfied. Similarly, post hoc testing for this alternative will use Games-Howell testing to decipher which pairs are statistically significant.

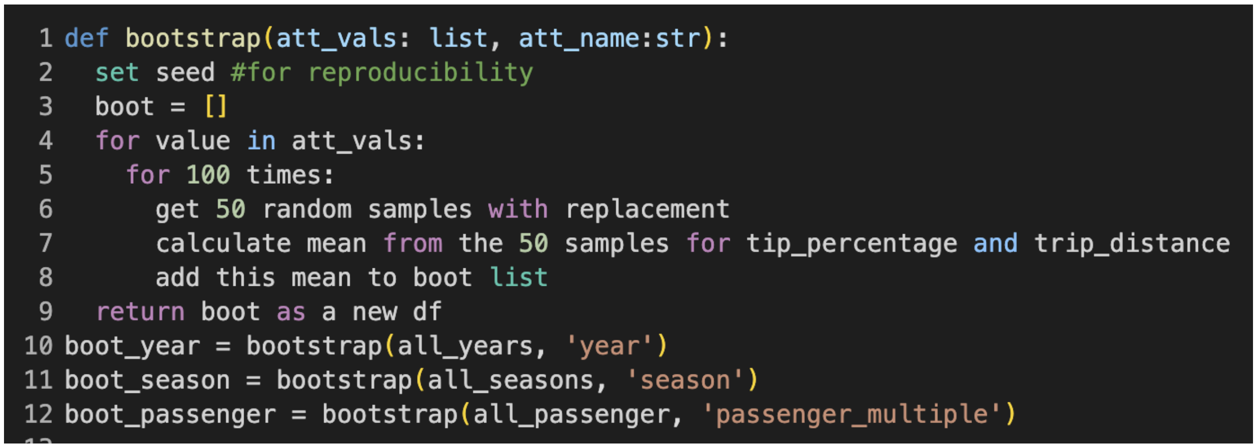
Unfortunately, the raw data itself does not follow a normal distribution in each group as seen in figure 1.1 below. To solve this, bootstrap sampling is used to invoke the Central Limit Theorem. The Central Limit Theorem states that sample means will reach an approximately normal distribution as the number of sample means reaches infinity and if the number of samples to create each mean is at least 30. Therefore, if bootstrap sampling is used to gather sample means, those sample means will reach a roughly normal distribution and will be treated as if it were the raw data for ANOVA. It is important to note that doing so will alter the interpretation of the ANOVA results slightly. The results will show whether a difference in the population mean of sample means are statistically significant rather than the population means of the raw data themselves are different. Nonetheless, the results will still reveal an interesting underlying pattern in the data.



```
Shapiro-Wilk test for 2009: p-value = 0.0 - NOT NORMAL DISTRIBUTION
Shapiro-Wilk test for 2011: p-value = 0.0 - NOT NORMAL DISTRIBUTION
Shapiro-Wilk test for 2019: p-value = 0.0 - NOT NORMAL DISTRIBUTION
Shapiro-Wilk test for 2020: p-value = 0.0 - NOT NORMAL DISTRIBUTION
Shapiro-Wilk test for 2023: p-value = 0.0 - NOT NORMAL DISTRIBUTION


Levene's test: p-value = 1.7455092808250402e-59 - NOT EQUAL VARIANCE
/usr/local/lib/python3.10/dist-packages/scipy/stats/_morestats.py:1882: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
```

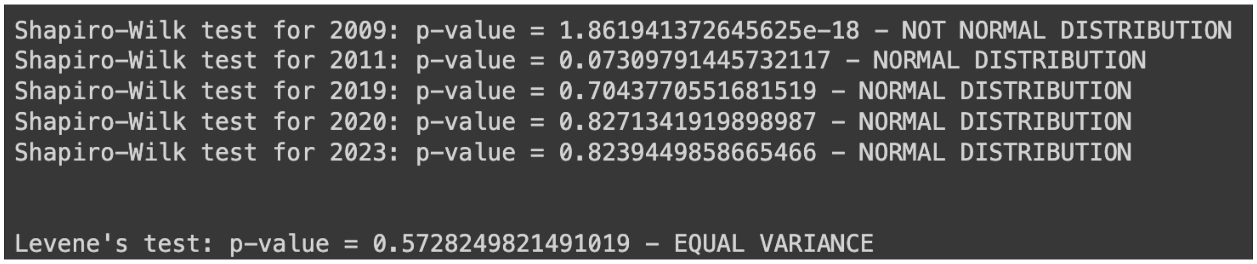
Figure 1.1 - Normality Test Prior to Bootstrapping w.r.t. ‘tip_percentage’

To perform bootstrapping sufficient enough to satisfy the Central Limit Theorem, we will randomly sample with replacement 50 instances from each group, calculate their mean, then repeat this 100 times for a total of 100 sample means per group. Not only will this allow the Central Limit Theorem to take place, but it will ensure independent samples while avoiding imbalanced group sizes. Pseudo-code for this process can be seen in figure 1.2. Once this is performed, all normality tests pass (excluding year 2009) as seen in figure 1.3. If one or two groups is not found to be normal, then post-hoc testing will continue, but results including those groups will be ignored.



```
def bootstrap(att_vals: list, att_name:str):
  set seed #for reproducibility
  boot = []
  for value in att_vals:
    for 100 times:
      get 50 random samples with replacement
      calculate mean from the 50 samples for tip_percentage and trip_distance
      add this mean to boot list
  return boot as a new df
boot_year = bootstrap(all_years, 'year')
boot_season = bootstrap(all_seasons, 'season')
boot_passenger = bootstrap(all_passenger, 'passenger_multiple')
```

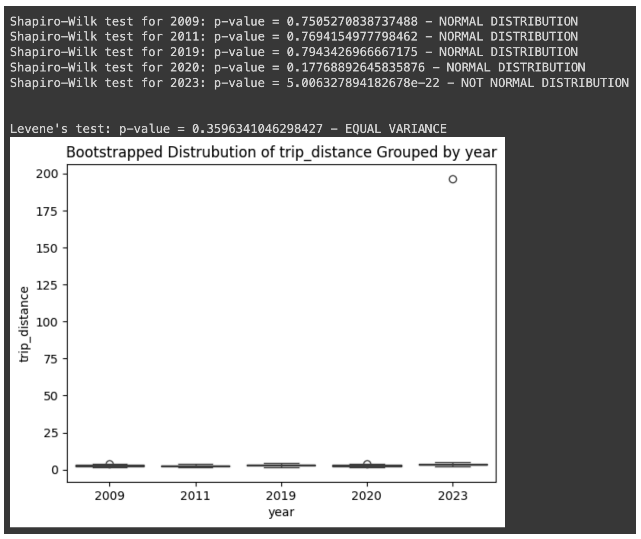
Figure 1.2 - Pseudo-Code for Bootstrapping Procedure



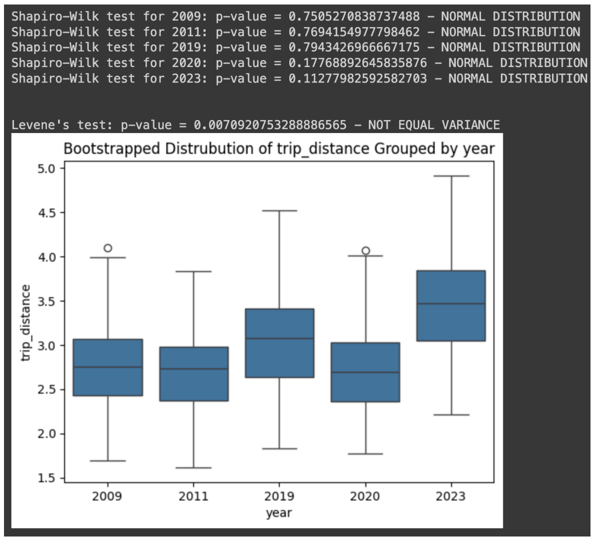
```
Shapiro-Wilk test for 2009: p-value = 1.861941372645625e-18 - NOT NORMAL DISTRIBUTION
Shapiro-Wilk test for 2011: p-value = 0.07309791445732117 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2019: p-value = 0.7043770551681519 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2020: p-value = 0.8271341919898987 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2023: p-value = 0.8239449858665466 - NORMAL DISTRIBUTION


Levene's test: p-value = 0.5728249821491019 - EQUAL VARIANCE
```

Figure 1.3 - Normality Testing After Bootstrapping w.r.t. ‘tip_percentage’

When bootstrapping for year, the distribution based on trip distance produces an extreme outlier leading to a non-normal distribution as seen in figure 1.4. When removing this outlier, this issue is fixed shown in figure 1.5.



```
Shapiro-Wilk test for 2009: p-value = 0.7505270838737488 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2011: p-value = 0.7694154977798462 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2019: p-value = 0.7943426966667175 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2020: p-value = 0.17768892645835876 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2023: p-value = 5.006327894182678e-22 - NOT NORMAL DISTRIBUTION


Levene's test: p-value = 0.3596341046298427 - EQUAL VARIANCE
```



```
Shapiro-Wilk test for 2009: p-value = 0.7505270838737488 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2011: p-value = 0.7694154977798462 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2019: p-value = 0.7943426966667175 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2020: p-value = 0.17768892645835876 - NORMAL DISTRIBUTION
Shapiro-Wilk test for 2023: p-value = 0.11277982592582703 - NORMAL DISTRIBUTION


Levene's test: p-value = 0.007092075328886565 - NOT EQUAL VARIANCE
```

Figure 1.4 (left) - Bootstrap Distribution & Figure 1.5 (right) - Bootstrap Distribution Outlier Removed

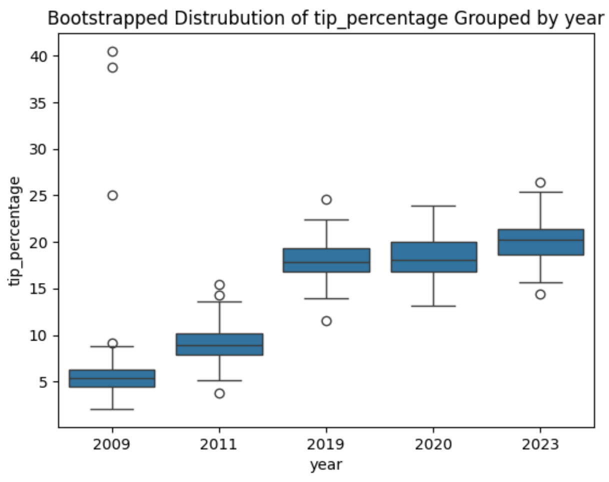
With these steps and the outlier adjustment mentioned previously, all groups now follow a normal distribution with equal variance with exceptions to tip percentage in 2009, and tip percentage for Winter and Spring. There is also unequal variance in trip distance for year, but this can be solved with Welch's ANOVA and Games-Howell. With these assumptions met, ANOVA can now be used reliably.

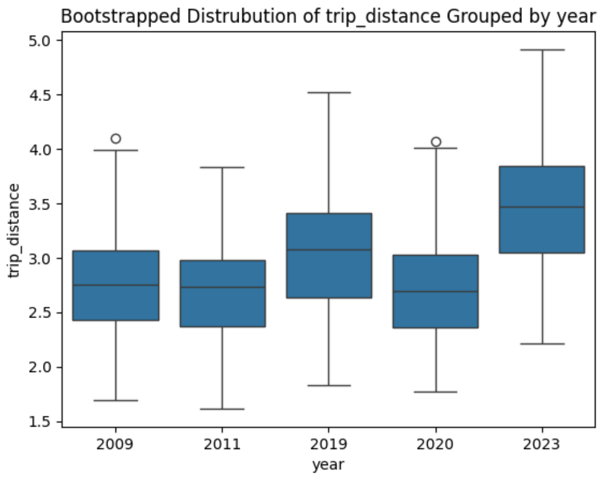
Evaluation Methodology

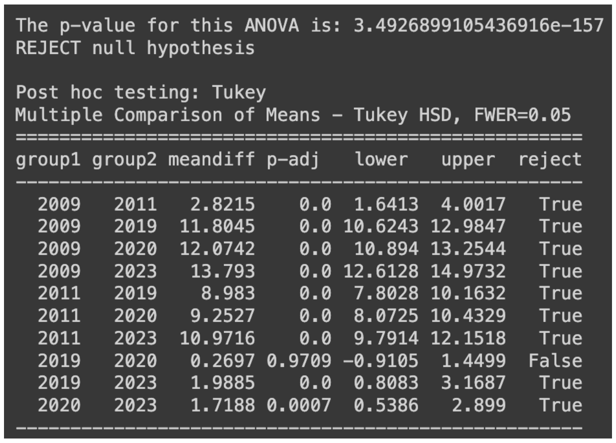
Because the focus of this report is not on prediction, there is no performance of evaluate, rather insights on to various patterns the data holds.

Results and Discussion

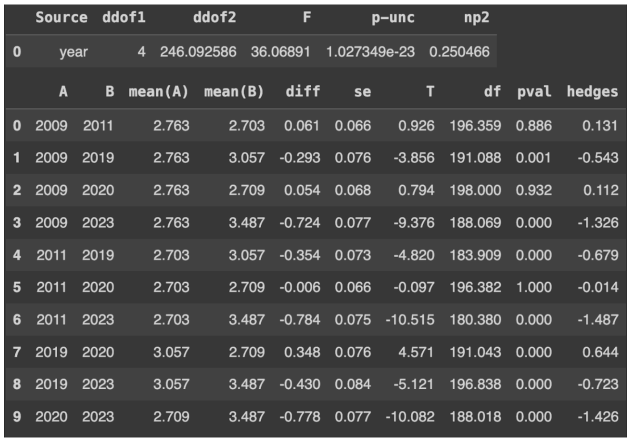
<u>ANOVA</u>







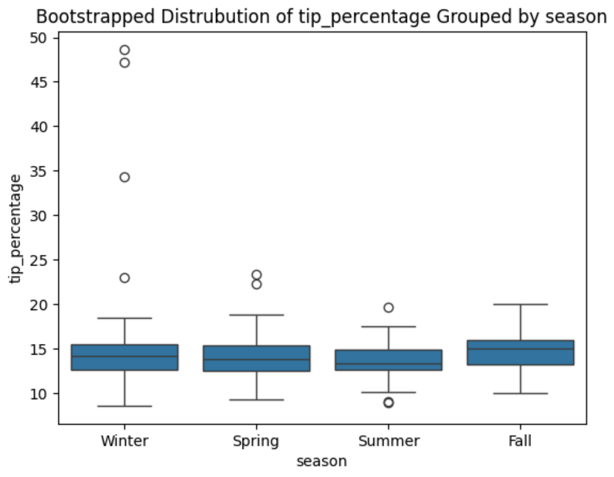
```
The p-value for this ANOVA is: 3.4926899105436916e-157
REJECT null hypothesis

Post hoc testing: Tukey
Multiple Comparison of Means - Tukey HSD, FWER=0.05
=====================================================
group1 group2 meandiff p-adj   lower   upper  reject
-----------------------------------------------------
  2009   2011   2.8215    0.0  1.6413  4.0017   True
  2009   2019  11.8045    0.0 10.6243 12.9847   True
  2009   2020  12.0742    0.0  10.894 13.2544   True
  2009   2023   13.793    0.0 12.6128 14.9732   True
  2011   2019    8.983    0.0  7.8028 10.1632   True
  2011   2020   9.2527    0.0  8.0725 10.4329   True
  2011   2023  10.9716    0.0  9.7914 12.1518   True
  2019   2020   0.2697 0.9709 -0.9105  1.4499  False
  2019   2023   1.9885    0.0  0.8083  3.1687   True
  2020   2023   1.7188 0.0007  0.5386   2.899   True
-----------------------------------------------------
```

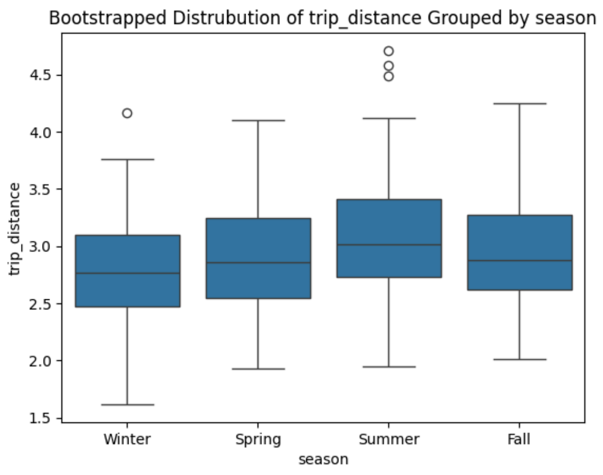


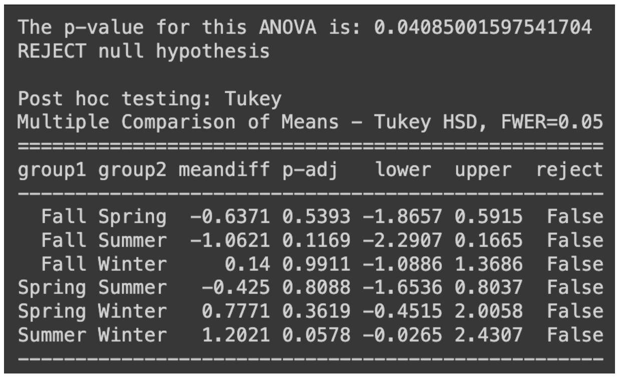
	Source	ddof1	ddof2	F	p-unc	np2
0	year	4	246.092586	36.06891	1.027349e-23	0.250466

	A	B	mean(A)	mean(B)	diff	se	T	df	pval	hedges
0	2009	2011	2.763	2.703	0.061	0.066	0.926	196.359	0.886	0.131
1	2009	2019	2.763	3.057	-0.293	0.076	-3.856	191.088	0.001	-0.543
2	2009	2020	2.763	2.709	0.054	0.068	0.794	198.000	0.932	0.112
3	2009	2023	2.763	3.487	-0.724	0.077	-9.376	188.069	0.000	-1.326
4	2011	2019	2.703	3.057	-0.354	0.073	-4.820	183.909	0.000	-0.679
5	2011	2020	2.703	2.709	-0.006	0.066	-0.097	196.382	1.000	-0.014
6	2011	2023	2.703	3.487	-0.784	0.075	-10.515	180.380	0.000	-1.487
7	2019	2020	3.057	2.709	0.348	0.076	4.571	191.043	0.000	0.644
8	2019	2023	3.057	3.487	-0.430	0.084	-5.121	196.838	0.000	-0.723
9	2020	2023	2.709	3.487	-0.778	0.077	-10.082	188.018	0.000	-1.426

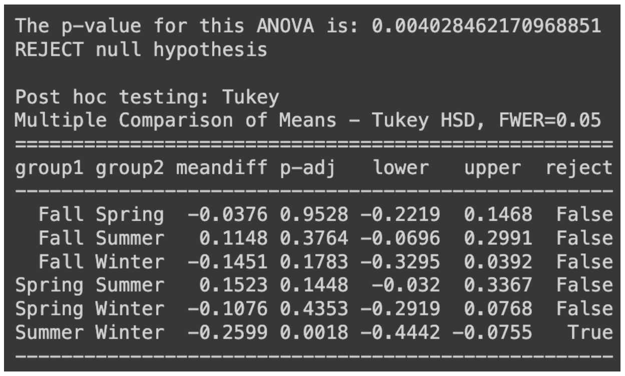
Figure 2.1 (left) & Figure 2.2 (right) - ANOVA Results for ‘year’ w.r.t. ‘tip_percentage’ (left) and ‘trip_distance’ (right)





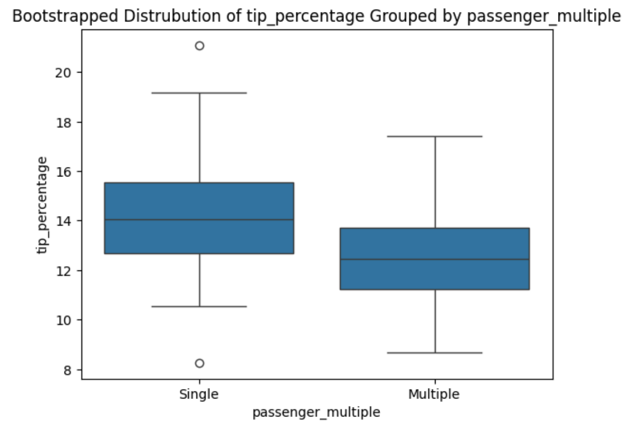


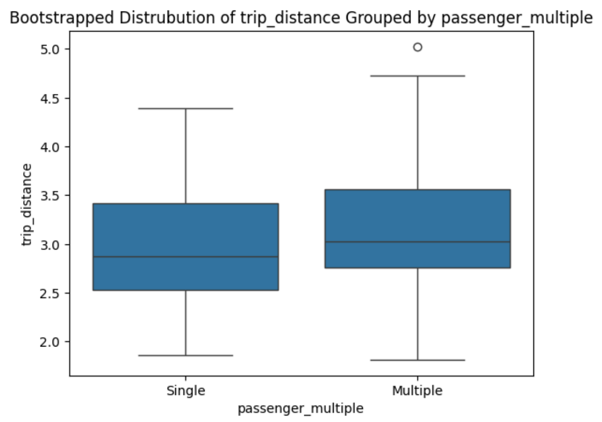
```
The p-value for this ANOVA is: 0.04085001597541704
REJECT null hypothesis

Post hoc testing: Tukey
Multiple Comparison of Means - Tukey HSD, FWER=0.05
=====================================================
group1 group2 meandiff p-adj   lower  upper  reject
-----------------------------------------------------
  Fall Spring  -0.6371 0.5393 -1.8657 0.5915  False
  Fall Summer  -1.0621 0.1169 -2.2907 0.1665  False
  Fall Winter     0.14 0.9911 -1.0886 1.3686  False
Spring Summer   -0.425 0.8088 -1.6536 0.8037  False
Spring Winter   0.7771 0.3619 -0.4515 2.0058  False
Summer Winter   1.2021 0.0578 -0.0265 2.4307  False
-----------------------------------------------------
```

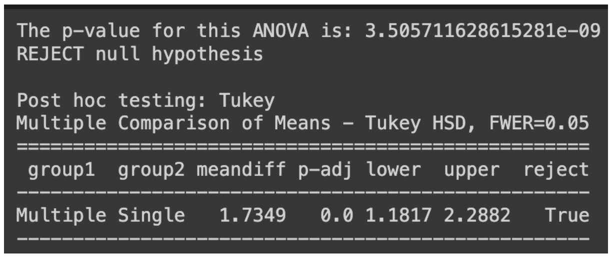


```
The p-value for this ANOVA is: 0.004028462170968851
REJECT null hypothesis

Post hoc testing: Tukey
Multiple Comparison of Means - Tukey HSD, FWER=0.05
=====================================================
group1 group2 meandiff p-adj   lower   upper  reject
-----------------------------------------------------
  Fall Spring  -0.0376 0.9528 -0.2219  0.1468  False
  Fall Summer   0.1148 0.3764 -0.0696  0.2991  False
  Fall Winter  -0.1451 0.1783 -0.3295  0.0392  False
Spring Summer   0.1523 0.1448  -0.032  0.3367  False
Spring Winter  -0.1076 0.4353 -0.2919  0.0768  False
Summer Winter  -0.2599 0.0018 -0.4442 -0.0755   True
-----------------------------------------------------
```

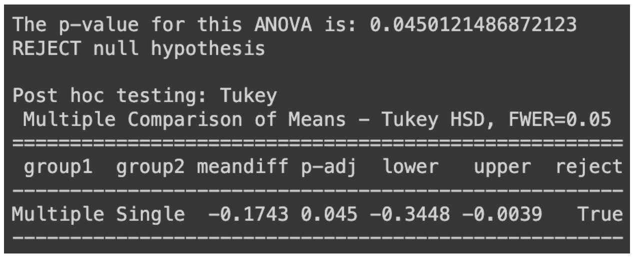
Figure 2.3 (left) & Figure 2.4 (right) - ANOVA Results for ‘season’ w.r.t. ‘tip_percentage’ (left) and ‘trip_distance’ (right)







```
The p-value for this ANOVA is: 3.505711628615281e-09
REJECT null hypothesis

Post hoc testing: Tukey
Multiple Comparison of Means - Tukey HSD, FWER=0.05
===================================================
 group1  group2 meandiff p-adj lower  upper  reject
---------------------------------------------------
Multiple Single   1.7349   0.0 1.1817 2.2882   True
---------------------------------------------------
```



```
The p-value for this ANOVA is: 0.0450121486872123
REJECT null hypothesis

Post hoc testing: Tukey
 Multiple Comparison of Means - Tukey HSD, FWER=0.05
=====================================================
 group1  group2 meandiff p-adj  lower   upper  reject
-----------------------------------------------------
Multiple Single  -0.1743 0.045 -0.3448 -0.0039   True
-----------------------------------------------------
```

Figure 2.5 (left) & Figure 2.6. (right) - ANOVA Results for ‘passenger_multiple’ w.r.t. ‘tip_percentage’ (left) and ‘trip_distance’ (right)

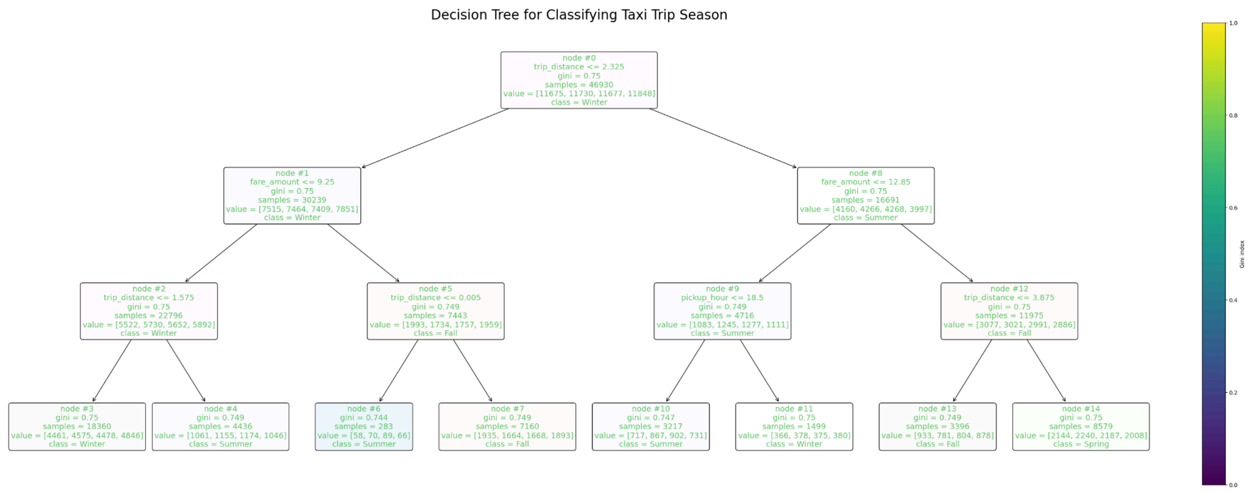
	Year (2009, 2011, 2019, 2020, 2023)	Season (Winter, Spring, Summer, Fall)	Passenger (Single or Multiple)
Tip Percentage	2011 & 2019 - (9.05 & 18.03) 2011 & 2020 - (9.05 & 18.03) 2011 & 2023 - (9.05 & 20.02) 2019 & 2023 - (18.03 & 20.02) 2020 & 2023 - (18.03 & 20.02)	No difference	Single & Multiple (14.20 & 12.46)
Trip Distance	2009 & 2019 - (2.76 & 3.06) 2009 & 2023 - (2.76 & 3.49) 2011 & 2019 - (2.70 & 3.06) 2011 & 2023 - (2.70 & 3.49) 2019 & 2020 - (3.06 & 2.71) 2019 & 2023 - (3.06 & 3.49) 2020 & 2023 - (2.71 & 3.49)	Summer & Winter (3.06 & 2.80)	Single & Multiple (2.98 & 3.15)

Table 2.1 - ANOVA Final Results Detailing Which Difference Pairs are Statistically Significant

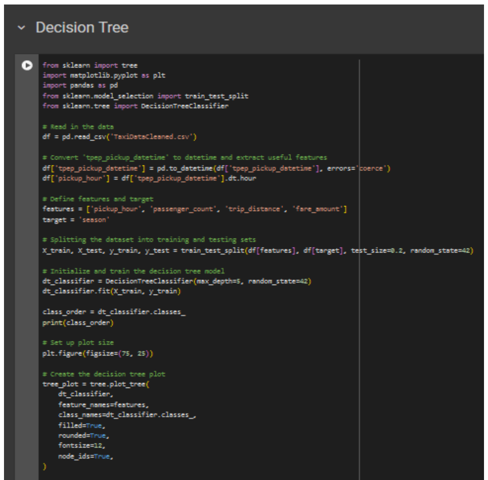
Given the final results for ANOVA testing, the initial questions can be answered. Firstly, seasons do not appear to truly impact tip percentage, but it may impact trip distance such that people are likely to travel longer distances in the summer than winter. This intuitively makes sense as the warmer summer days may inspire people to travel to further places. Secondly, the number of passengers (whether they are alone or with multiple people) seems to appear statistically significant for both tip percentage and trip distance; single passengers are likely to tip higher while traveling less distance as opposed to multiple passengers leaning towards lower tip percentage and higher travel distances. The higher tip percentage when riding alone may be caused by more personal social pressure to tip higher when one is alone with the driver. In contrast, a party of passengers traveling further distances may be due to social events being located further away. Thirdly, Covid did not appear to impact tip percentage after 2019, however, tipping did go up since 2020. In terms of trip distance, the average distance is lower in 2020 than in 2019 and 2023 as to be expected from many places being shut down. Fourthly, there is no difference in tips or trip distance between 2009 and 2011 implying that the introduction of Uber did not immediately alter anything. There is a difference between 2011 and later years such as 2019 meaning that both tipping and trip distance have increased since then. However, because the gap between 2011 and 2019 is so large, it is hard to distinguish whether Uber had a role in this change.

ANOVA testing provided some insightful results about the underlying nature of the data. While bootstrapping had to be used to be able to trust ANOVA's results, the findings still suggest a general trend among NYC taxi data. There is a slight issue when considering the impact of Uber, and that is the large gap between the introduction of Uber to NYC in 2011 and the next available year 2019. While there was a significant difference to be found after Uber had been integrated, this gap is too large to isolate the effects of Uber. If further analysis were to be performed, adding another year closer to 2012 may yield more beneficial results.

<u>DECISION TREE</u>



Our decision tree shows important quantifiers to help understand our data. Our primary target for values were seasons, because we are doing an analysis of seasonal trends. Throughout the tree we are tracking the Gini index which should be near .75 because there are 4 seasons. Some of the features we decided to take a look at how much money is made, what time the trip occurs, distance, and how many people.



```
from sklearn import tree
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier

# Read in the data
df = pd.read_csv('TaxiDataCleaned.csv')

# Convert 'tpep_pickup_datetime' to datetime and extract useful features
df['tpep_pickup_datetime'] = pd.to_datetime(df['tpep_pickup_datetime'], errors='coerce')
df['pickup_hour'] = df['tpep_pickup_datetime'].dt.hour

# Define features and target
features = ['pickup_hour', 'passenger_count', 'trip_distance', 'fare_amount']
target = 'season'

# Splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(df[features], df[target], test_size=0.2, random_state=42)

# Initialize and train the decision tree model
dt_classifier = DecisionTreeClassifier(max_depth=5, random_state=42)
dt_classifier.fit(X_train, y_train)

class_order = dt_classifier.classes_
print(class_order)

# Set up plot size
plt.figure(figsize=(75, 25))

# Create the decision tree plot
tree_plot = tree.plot_tree(
    dt_classifier,
    feature_names=features,
    class_names=dt_classifier.classes_,
    filled=True,
    rounded=True,
    fontsize=12,
    node_ids=True,
)
```

We see at node #0 that the trip distance is the most common identifier to be split, when the miles are greater or less then 2.325 being the classifier.

In node #1 we if the fare amount is less than or equal to $9.25 then in node#8 being the opposing node from 1 stemming from node 0, we see that the fare is typically greater than $12.85.

After observing all the nodes we are able to see microtrends such as that the most canceled rides are around fall going into winter time. In Node #5, we see mostly canceled trips documented under $9.25, and under .005 miles (26 feet).

Looking at node #9, we can see another trend that most pickups during summer are occurring later in the day around 6:30pm.

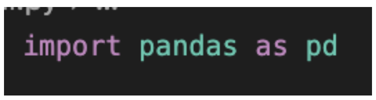
At the end, this is important to observe seasonal business planning, for taxi companies to adjust strategies according to season, help adjust pricing per demand, and help become more efficient.

PEARSON CORRELATION COEFFICIENT

The Pearson correlation coefficient is a measure of the strength and direction of the linear relationship between two variables. It ranges from -1 to 1, where 1 indicates a perfect positive linear relationship, -1 indicates a perfect negative linear relationship, and 0 indicates no linear relationship. In our case there are quite a bit of value we want to compare to answer the question established at the beginning of the report. The desired comparisons are:

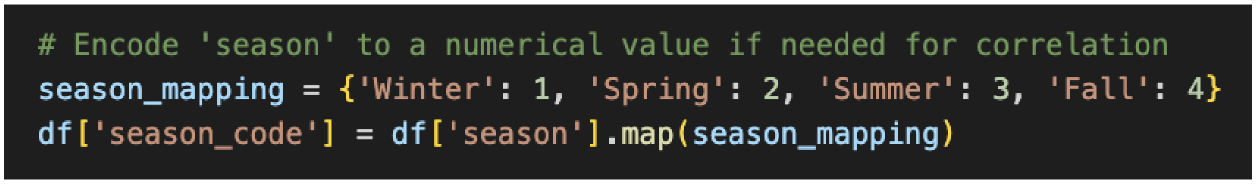
- Season x Taxi Frequency
- Season x Tip Amount
- Season x Taxi Distance
- # of Passengers x Tips
- # of Passengers x Distance
- Tip Amount x Distances
- Tip Amount x Fair Amount

To accomplish these calculations I first imported the pandas library:



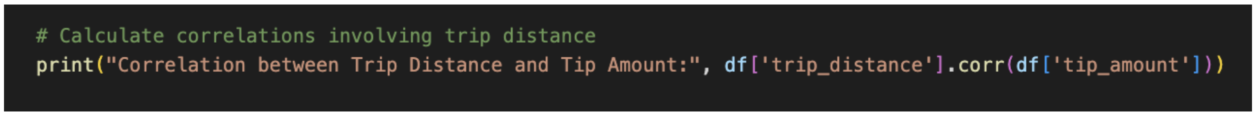
```
import pandas as pd
```

Then I created a numerical value for each season:



```
# Encode 'season' to a numerical value if needed for correlation
season_mapping = {'Winter': 1, 'Spring': 2, 'Summer': 3, 'Fall': 4}
df['season_code'] = df['season'].map(season_mapping)
```

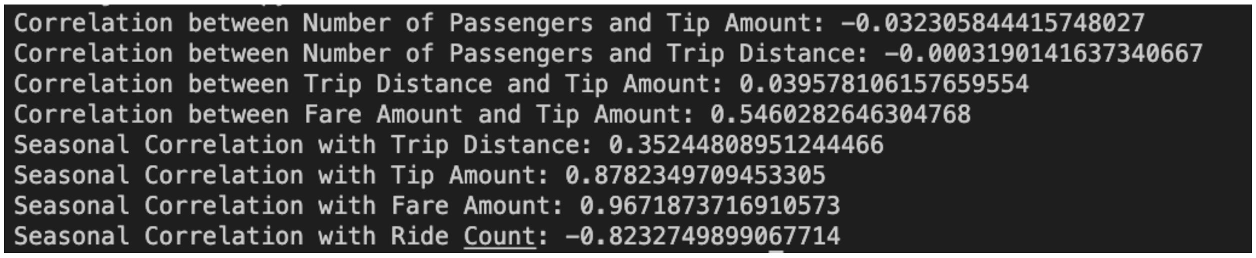
Then I used the '.corr' function to calculate the correlations:



```
# Calculate correlations involving trip distance
print("Correlation between Trip Distance and Tip Amount:", df['trip_distance'].corr(df['tip_amount']))
```

Using this function I made similar print statements to the one above for each comparison.

Doing this for each comparison proved these results:



```
Correlation between Number of Passengers and Tip Amount: -0.032305844415748027
Correlation between Number of Passengers and Trip Distance: -0.0003190141637340667
Correlation between Trip Distance and Tip Amount: 0.039578106157659554
Correlation between Fare Amount and Tip Amount: 0.5460282646304768
Seasonal Correlation with Trip Distance: 0.35244808951244466
Seasonal Correlation with Tip Amount: 0.8782349709453305
Seasonal Correlation with Fare Amount: 0.9671873716910573
Seasonal Correlation with Ride Count: -0.8232749899067714
```

Results:

- Season x Taxi Frequency:

The correlation analysis for the season x taxi frequency was -.8233. This means there was a high negative linear correlation. In other words, as the season went from winter to fall, the amount of taxi rides decreased in a strong linear fashion.

- Season x Tip Amount

The correlation analysis for the season x tip amount was 0.8782. This means there was a strong linear correlation, or, as we go from winter to fall the amount tipped increased.

- Season x Taxi Distance

The correlation analysis for the season x taxi distance was 0.3524. This is a very weak positive linear correlation. In other words, as we go from winter to fall the amount tipped is not very affected.

- # of Passengers x Tipss

The correlation analysis for the number of passengers x Tips produced a coefficient of -0.0323. This number represents a very weak correlation. This means that the number of tips does not correlate with the number of passengers.

- # of Passengers x Distance

The correlation analysis of the number of passengers x distance produced a coefficient of -0.0003. This is an extremely weak correlation and means the number of passengers does not affect the trip distance.

- Tip Amount x Distances

The correlation analysis of the tip amount x trip distance was 0.0396. This represents a very weak positive linear correlation. This means the distance had little correlation with the amount tipped.

- Tip Amount x Fair Amount

The correlation analysis of the tip amount x fair amount provided a coefficient of 0.5460. This coefficient represents a medium strength positive linear correlation. In other words, as the tip amount went up the fair amount also went up.