Uber Jan-Feb-FOIL Data Analytics

Presented by: Bruna Augustin

Agenda

1. Introduction
2. Why Data Analysis?
3. Project Focus
4. Dataset Overview
5. Data Cleaning and Processing
6. Data Visualization
7. Challenge Faced
8. Conclusion

Introduction

Uber is a leading ride-sharing platform that operates through a network of dispatching bases. This project analyzes Uber ride data focusing on active vehicles, total trips, and time-based patterns such as daily and monthly trends. The goal is to uncover operational insights that can help optimize resource allocation and improve ride efficiency

Why Data Analysis?

1. Identifying peak ride days and months for operational planning.

2. Understanding the trend of active vehicles and trips over time.

3. Analyzing weekday and monthly patterns to optimize resource allocation.

Project Focus

1. Data Cleaning & Preprocessing (Fixing missing or inconsistent values).

2. Trip and Active Vehicle Analysis (Understanding operational scale).

3. Weekday & Month Trend Analysis (Finding patterns in ride data).

Dataset Overview

Dataset Name – Uber Jan-Feb-FOIL
Source - Unified Mentor (Internship Project)
Total Records – 355 rows and 6 Column
Columns Included – dispatching_base_number, date, active vehicles, trips, weekday, month
Purpose – Analysing trip patterns, vehicle activity, and identifying operational trends

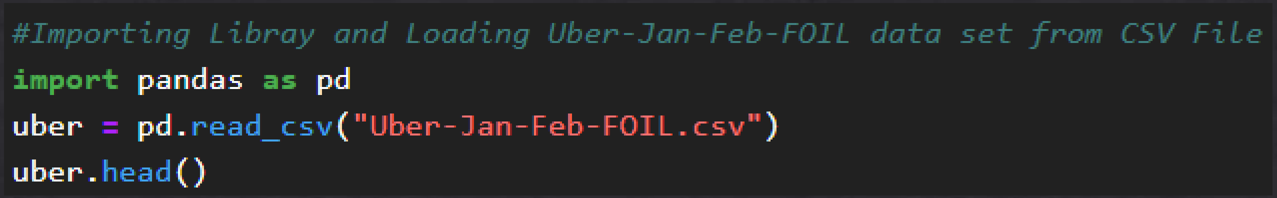
"This dataset, provided as part of the internship project, contains detailed records of Uber rides including dispatching base number, date, active vehicles, number of trips, weekday, and month information. The analysis focuses on uncovering ride frequency patterns, active vehicle trends, and operational insights using Python"

Data Cleaning and Processing With Python

We first loaded the Uber dataset and checked its structure using .info(). After handling missing values and removing duplicates, we converted the date column into DateTime format.

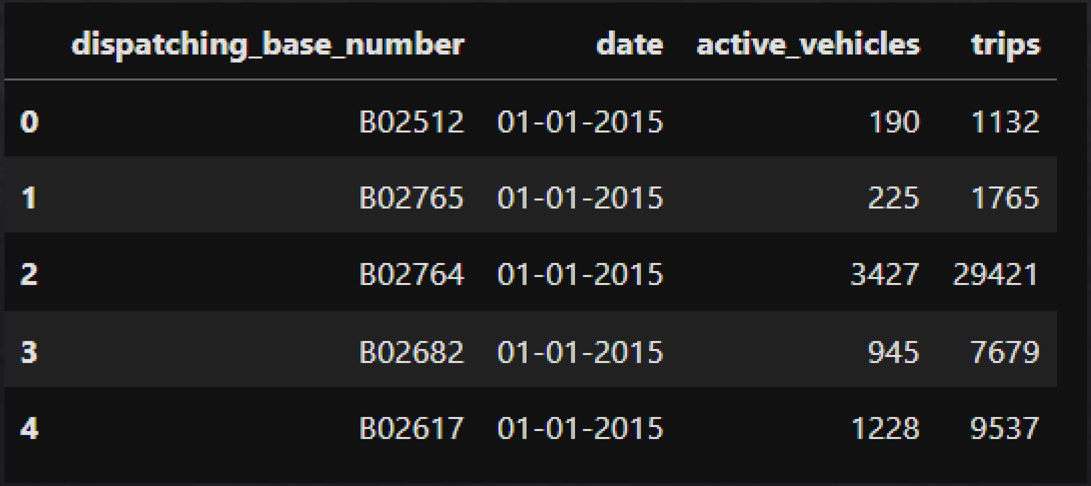
Finally, we extracted the weekday and month from the date to support time-based analysis.

Import Library and Loading Dataset



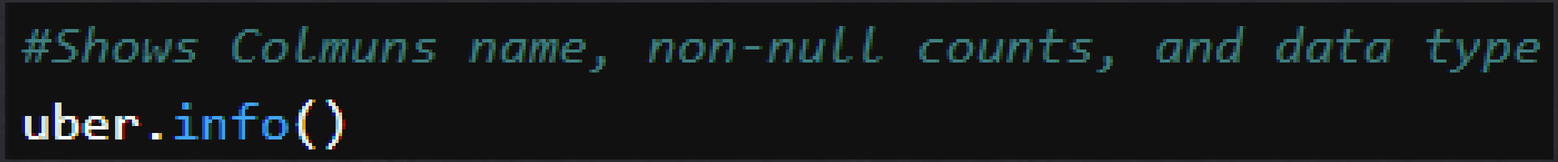
```
#Importing Libray and Loading Uber-Jan-Feb-FOIL data set from CSV File
import pandas as pd
uber = pd.read_csv("Uber-Jan-Feb-FOIL.csv")
uber.head()
```

Output



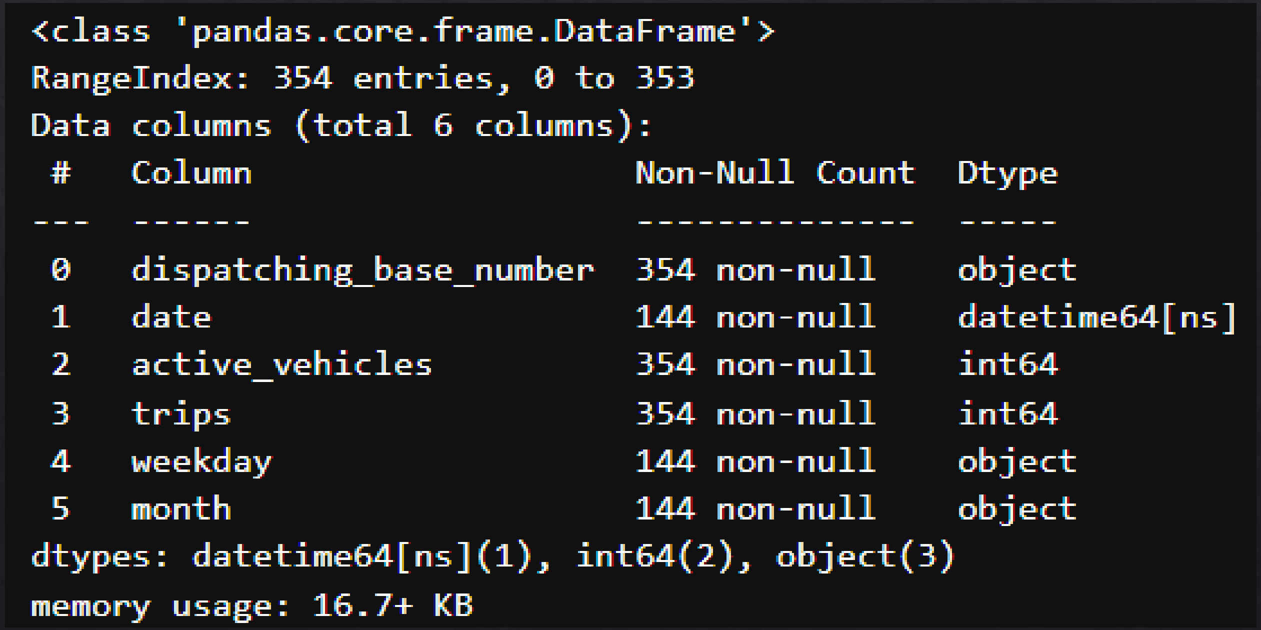
	dispatching_base_number	date	active_vehicles	trips
0	B02512	01-01-2015	190	1132
1	B02765	01-01-2015	225	1765
2	B02764	01-01-2015	3427	29421
3	B02682	01-01-2015	945	7679
4	B02617	01-01-2015	1228	9537

Checking Data Types & Structure



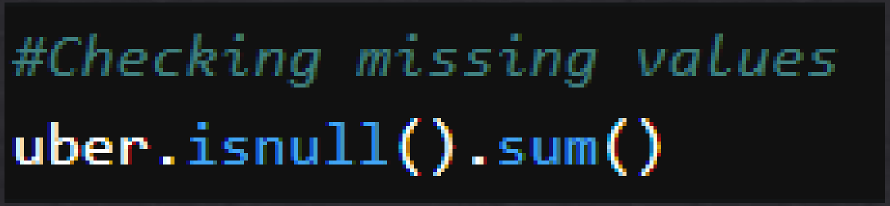
```python
#Shows Colmuns name, non-null counts, and data type
uber.info()
```

Output



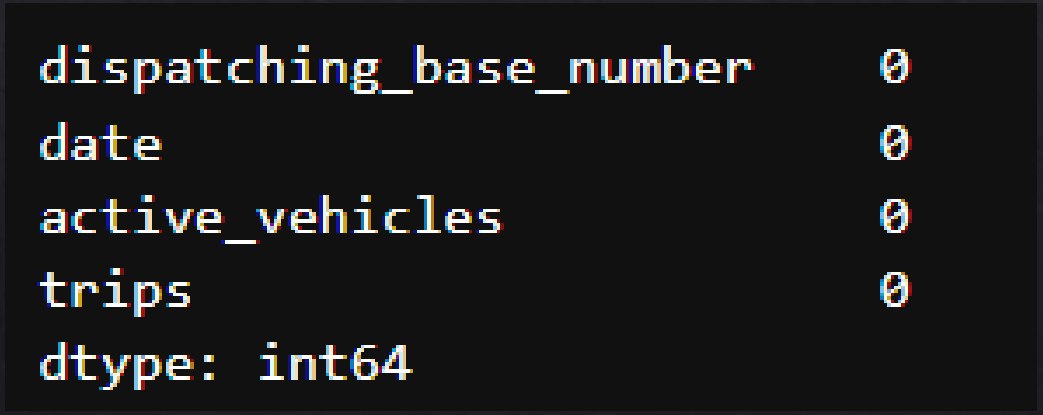
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 354 entries, 0 to 353
Data columns (total 6 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   dispatching_base_number  354 non-null    object
 1   date                     144 non-null    datetime64[ns]
 2   active_vehicles          354 non-null    int64
 3   trips                    354 non-null    int64
 4   weekday                  144 non-null    object
 5   month                    144 non-null    object
dtypes: datetime64[ns](1), int64(2), object(3)
memory usage: 16.7+ KB
```

Checking Missing Values



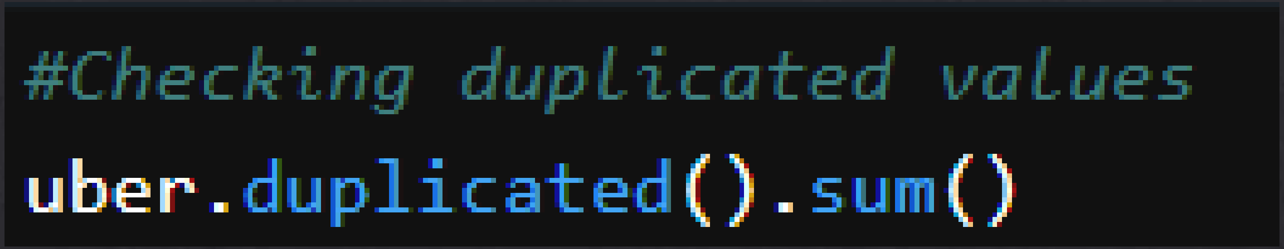
```python
#Checking missing values
uber.isnull().sum()
```

Output



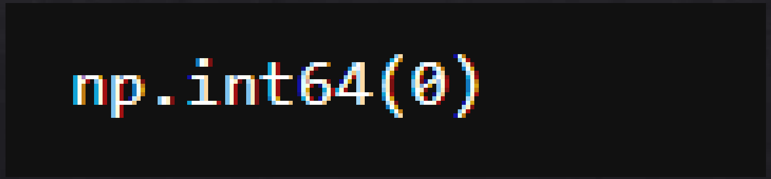
```
dispatching_base_number    0
date                       0
active_vehicles            0
trips                      0
dtype: int64
```

Checking Duplicates Values



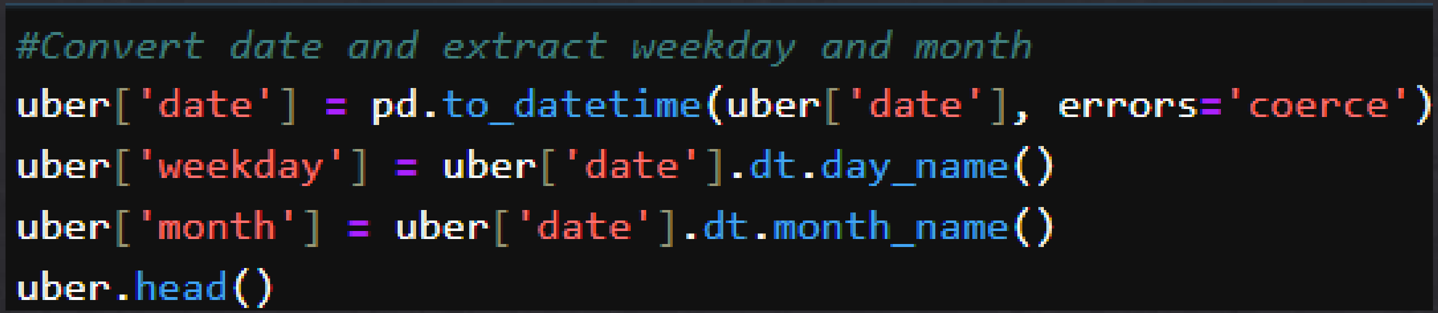
```
#Checking duplicated values
uber.duplicated().sum()
```

Output



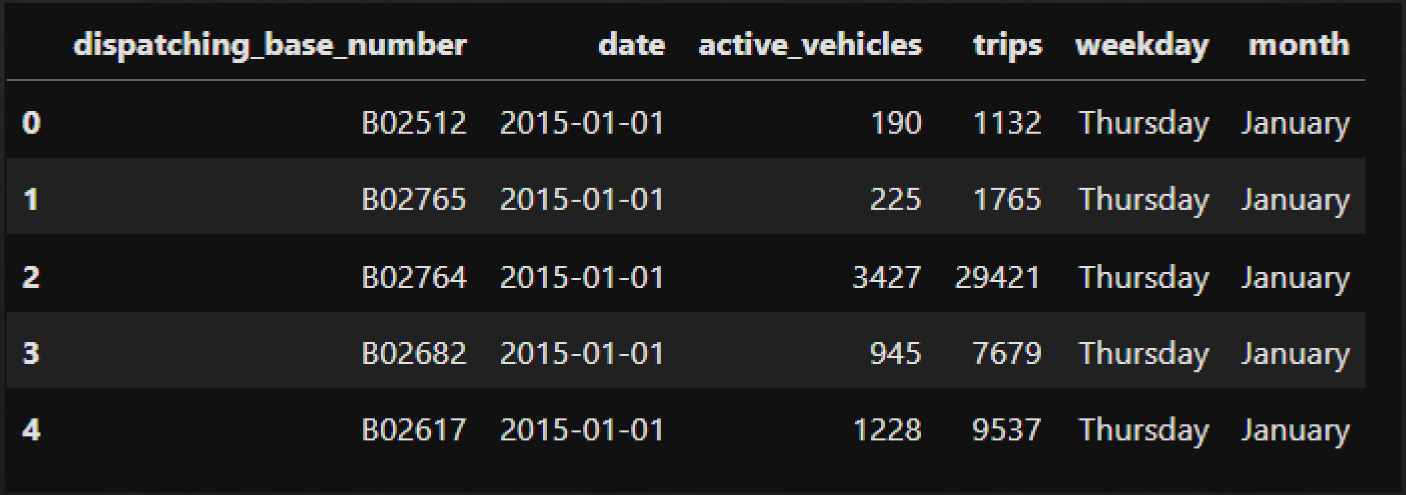
```
np.int64(0)
```

Convert Date and Extract weekday and Month



```
#Convert date and extract weekday and month
uber['date'] = pd.to_datetime(uber['date'], errors='coerce')
uber['weekday'] = uber['date'].dt.day_name()
uber['month'] = uber['date'].dt.month_name()
uber.head()
```

Output



	dispatching_base_number	date	active_vehicles	trips	weekday	month
0	B02512	2015-01-01	190	1132	Thursday	January
1	B02765	2015-01-01	225	1765	Thursday	January
2	B02764	2015-01-01	3427	29421	Thursday	January
3	B02682	2015-01-01	945	7679	Thursday	January
4	B02617	2015-01-01	1228	9537	Thursday	January

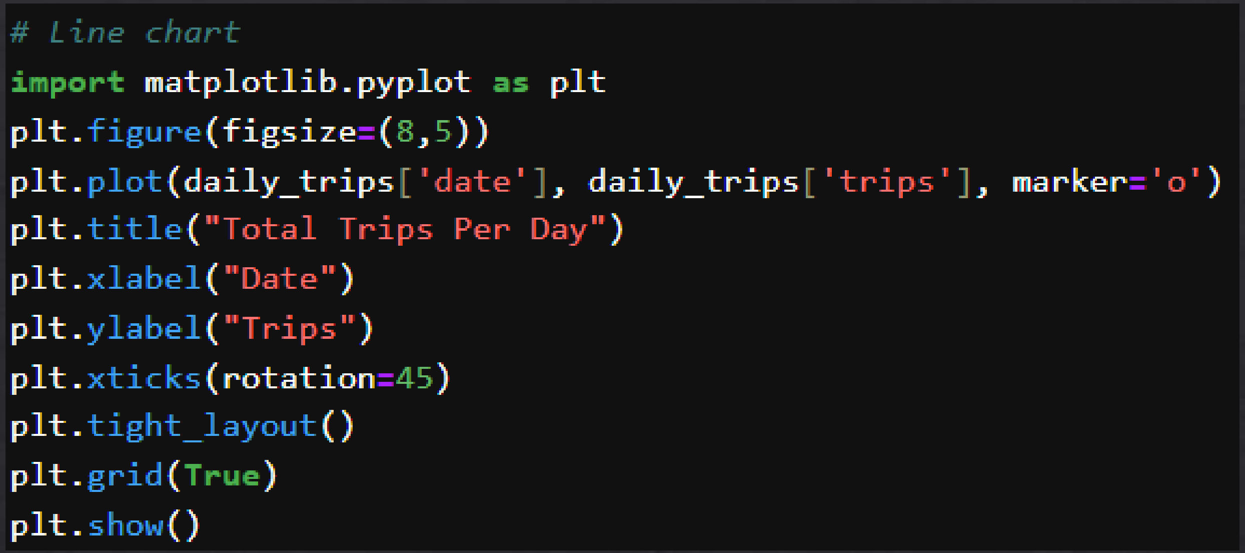
Data Visualization

After preparing the data, we created various visualizations to identify key patterns and insights.

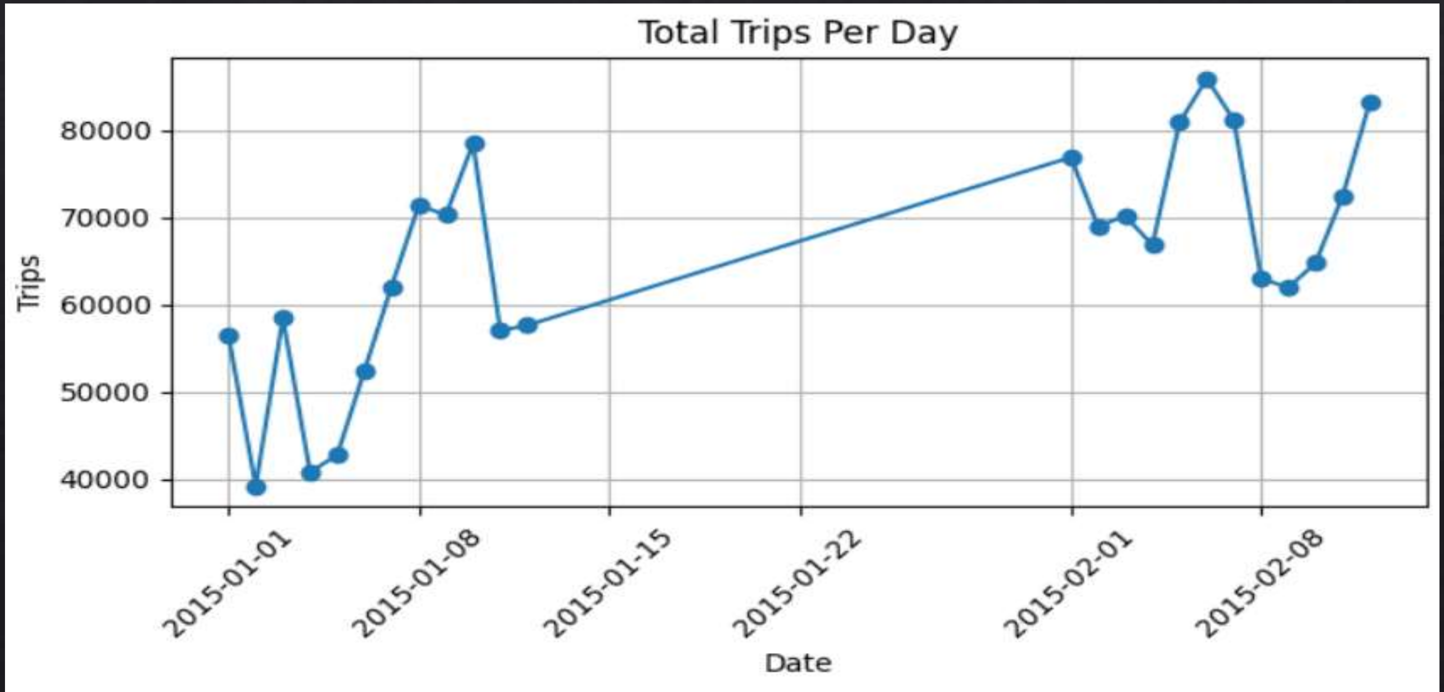
We used line charts to show trends over time, bar charts to compare trip and vehicle counts, pie charts and donut charts to represent distributions, and scatter plots to study the relationship between trips and active vehicles.

These visualizations made the data easier to interpret, analyze, and explain important trends effectively.

Line Chart

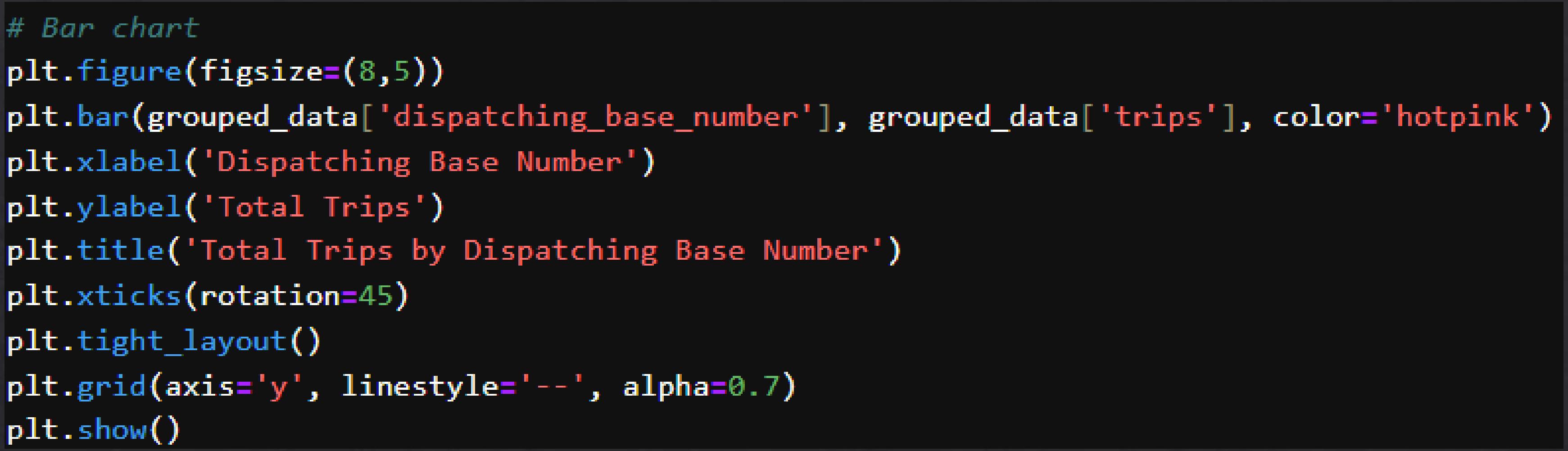


```
# Line chart
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
plt.plot(daily_trips['date'], daily_trips['trips'], marker='o')
plt.title("Total Trips Per Day")
plt.xlabel("Date")
plt.ylabel("Trips")
plt.xticks(rotation=45)
plt.tight_layout()
plt.grid(True)
plt.show()
```

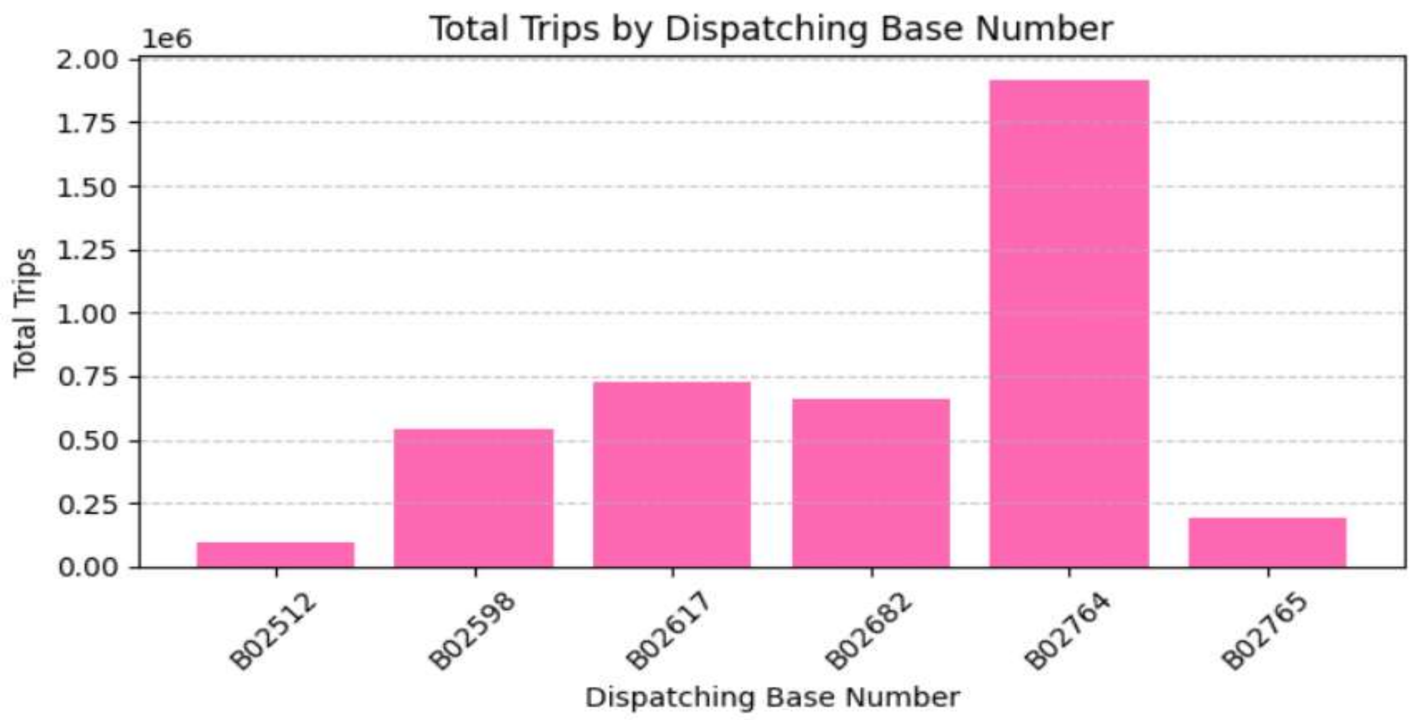


The chart displays daily trip counts, showing how the number of Uber rides changes over different dates.

Bar Chart

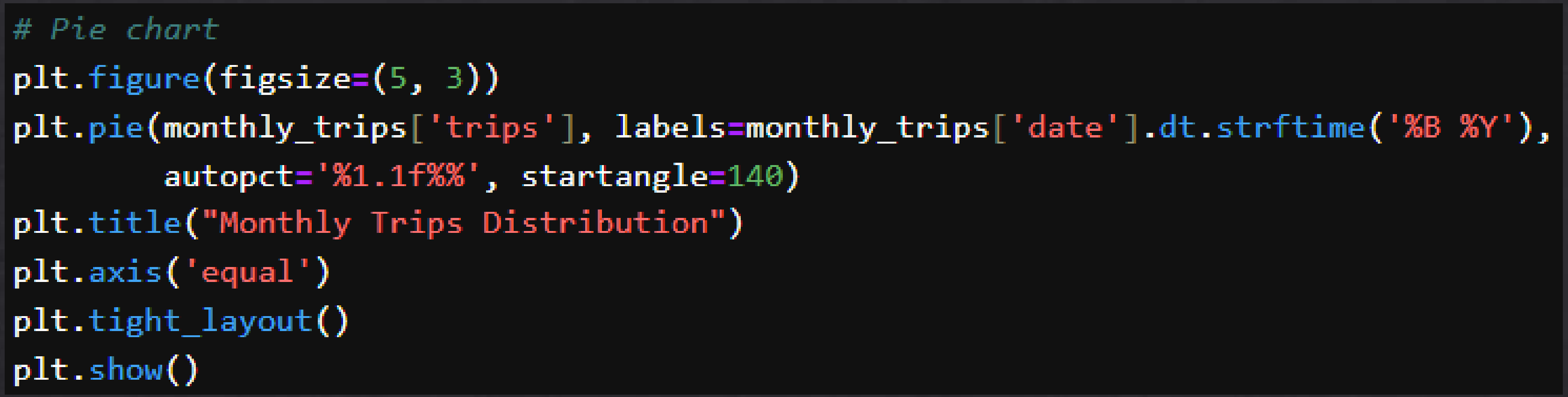


```python
# Bar chart
plt.figure(figsize=(8,5))
plt.bar(grouped_data['dispatching_base_number'], grouped_data['trips'], color='hotpink')
plt.xlabel('Dispatching Base Number')
plt.ylabel('Total Trips')
plt.title('Total Trips by Dispatching Base Number')
plt.xticks(rotation=45)
plt.tight_layout()
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

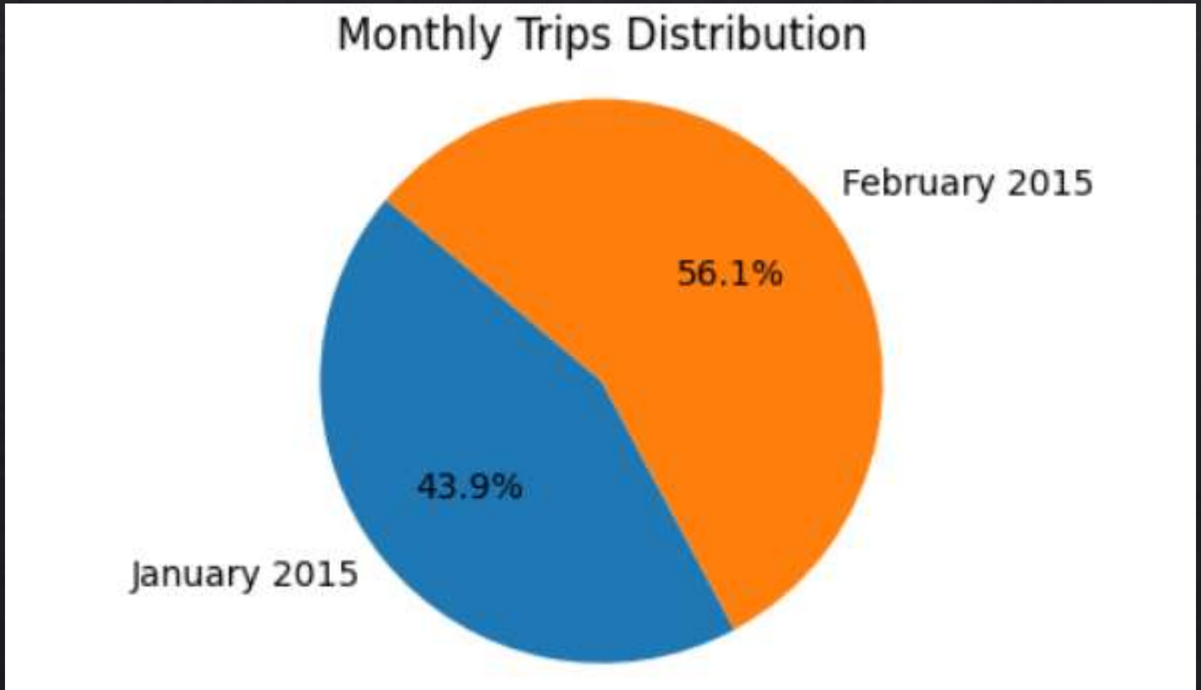


The chart compares total trips completed by each dispatch base, making it easy to see which base was busier

Pie Chart

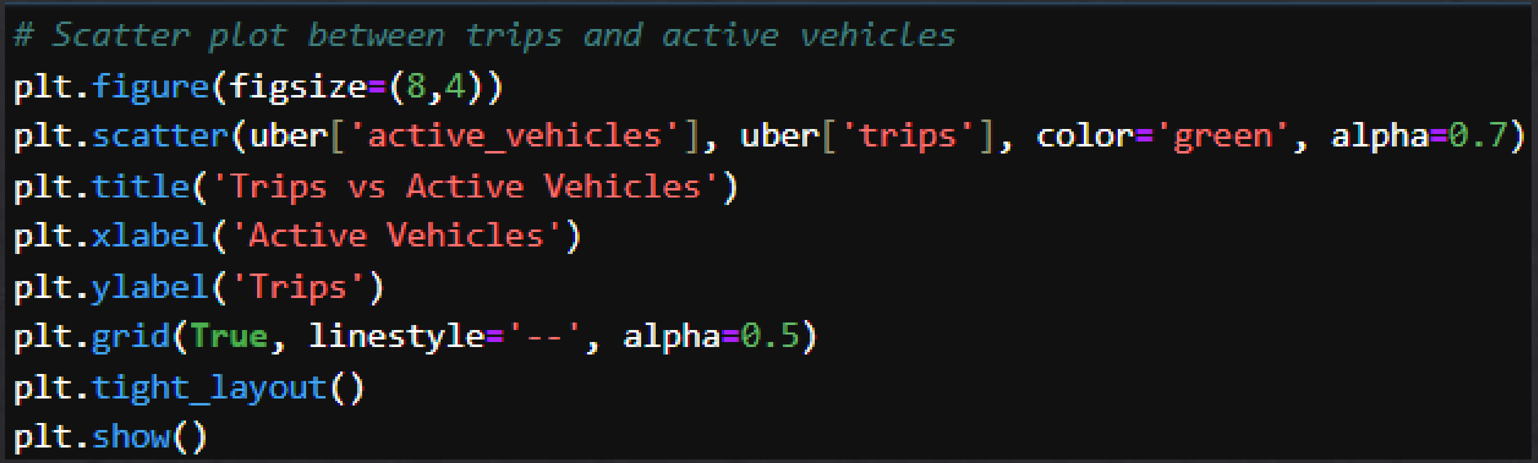


```
# Pie chart
plt.figure(figsize=(5, 3))
plt.pie(monthly_trips['trips'], labels=monthly_trips['date'].dt.strftime('%B %Y'),
        autopct='%1.1f%%', startangle=140)
plt.title("Monthly Trips Distribution")
plt.axis('equal')
plt.tight_layout()
plt.show()
```

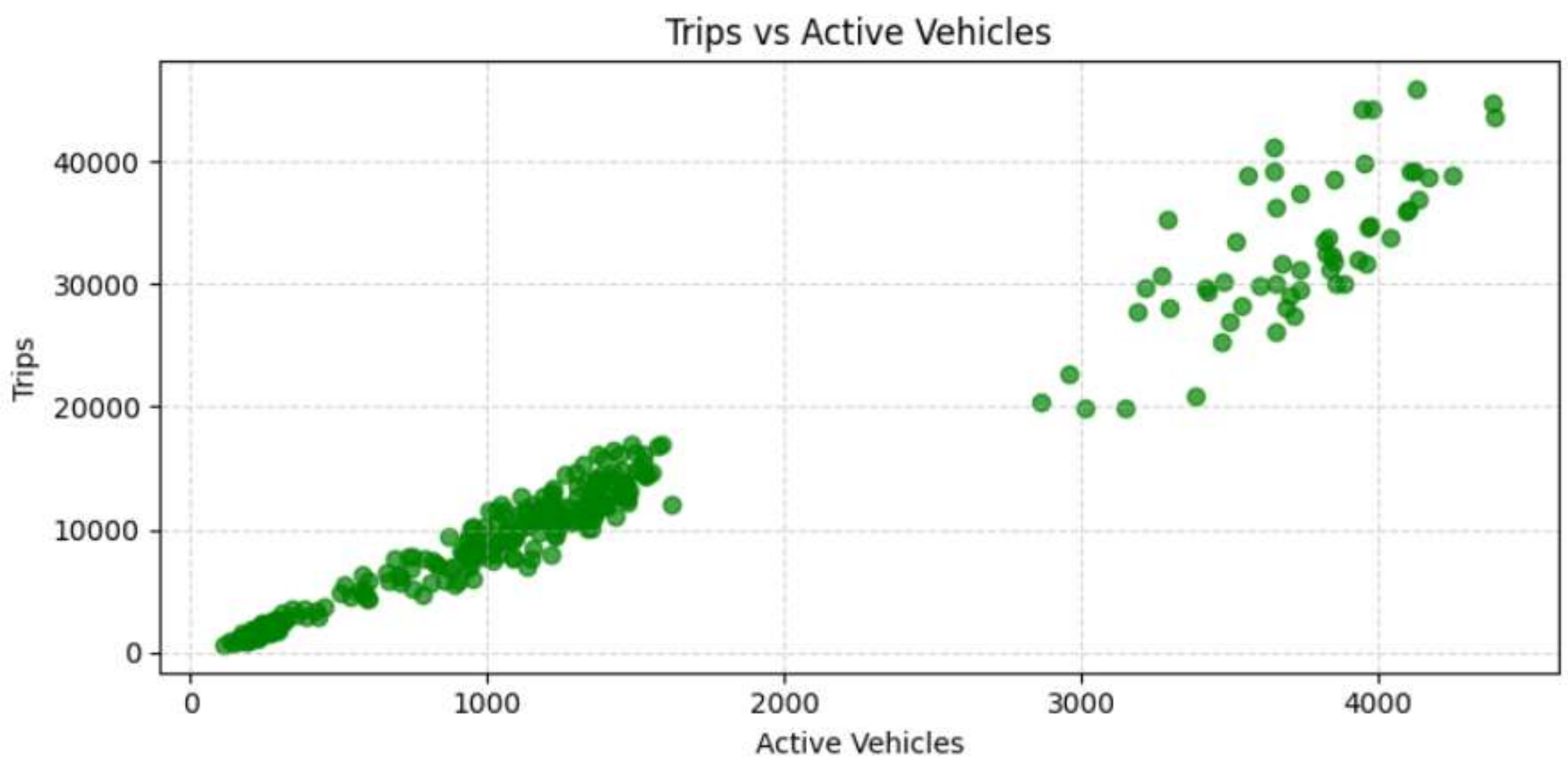


This pie chart shows the distribution of Uber trips across different months. It helps to visualize which months had higher or lower ride activity.

Scatter plot

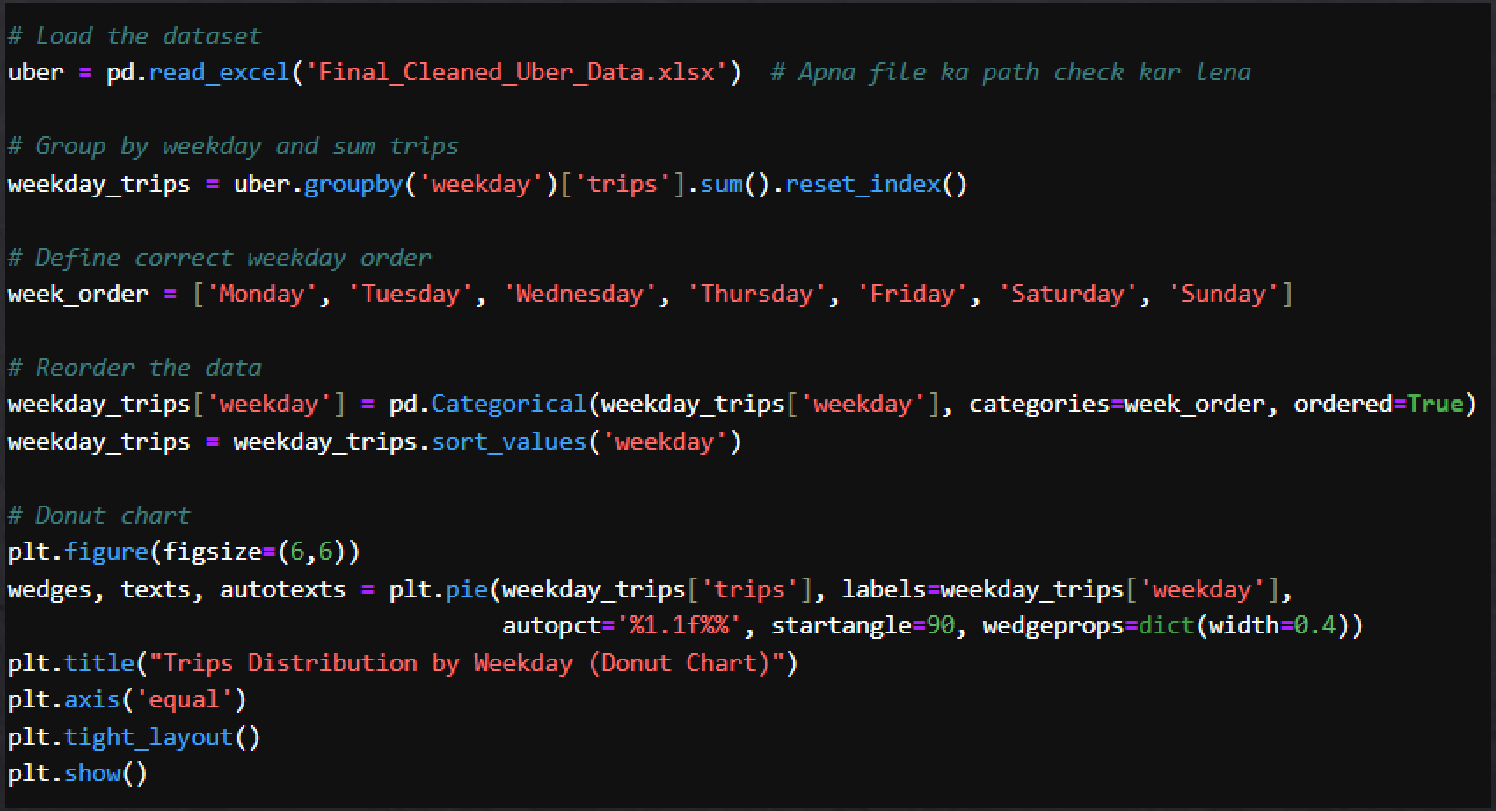


```
# Scatter plot between trips and active vehicles
plt.figure(figsize=(8,4))
plt.scatter(uber['active_vehicles'], uber['trips'], color='green', alpha=0.7)
plt.title('Trips vs Active Vehicles')
plt.xlabel('Active Vehicles')
plt.ylabel('Trips')
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



This scatter plot shows the relationship between active vehicles and the number of trips. It helps to understand how the number of active vehicles affects the total trips made.

Donut plot



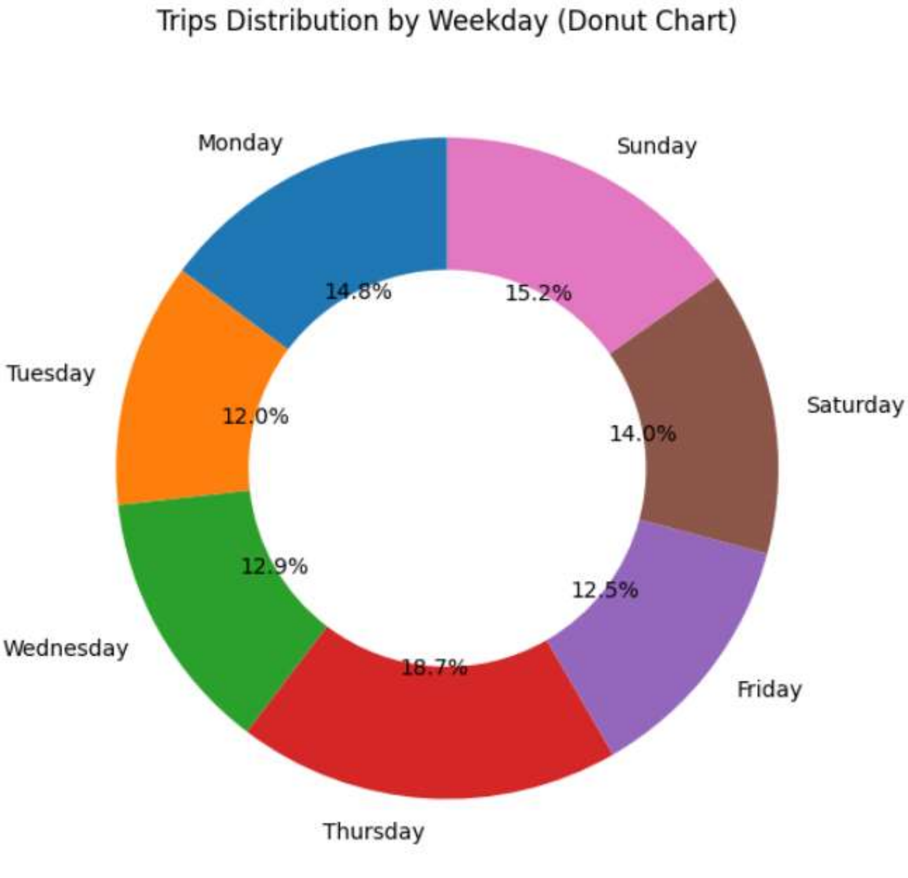
```
# Load the dataset
uber = pd.read_excel('Final_Cleaned_Uber_Data.xlsx')  # Apna file ka path check kar lena

# Group by weekday and sum trips
weekday_trips = uber.groupby('weekday')['trips'].sum().reset_index()

# Define correct weekday order
week_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']

# Reorder the data
weekday_trips['weekday'] = pd.Categorical(weekday_trips['weekday'], categories=week_order, ordered=True)
weekday_trips = weekday_trips.sort_values('weekday')

# Donut chart
plt.figure(figsize=(6,6))
wedges, texts, autotexts = plt.pie(weekday_trips['trips'], labels=weekday_trips['weekday'],
                                   autopct='%1.1f%%', startangle=90, wedgeprops=dict(width=0.4))
plt.title("Trips Distribution by Weekday (Donut Chart)")
plt.axis('equal')
plt.tight_layout()
plt.show()
```



This donut chart shows the distribution of trips across different weekdays. It highlights which days had higher or lower ride activity throughout the week.

Challenge Faced

1. Handling missing or invalid values to ensure clean and reliable data for accurate analysis.

2. Converting the 'date' column into a proper DateTime format to enable time-based insights.

3. Extracting correct weekday and month names from the date field for better trend analysis.

4. Grouping and sorting data properly to create meaningful and easy-to-read visualizations.

5. Maintaining the correct weekday order (Monday to Sunday) while creating charts for clear interpretation.

Conclusion

1. The Uber dataset was successfully cleaned by handling missing values and fixing date formats.

2. We extracted weekday and month features to analyze trip patterns more effectively.

3. Data visualizations like bar charts, pie charts, and scatter plots helped in understanding trends clearly.

4. Most trips were concentrated on specific weekdays and months, showing clear peak patterns.

5. Proper ordering and grouping of data provided more meaningful and accurate insights.

THANK YOU