Introduction:

In today's rapidly urbanizing world, understanding commuting patterns and behaviors is crucial for city planners, transportation authorities, and individuals alike. As cities grow, so do the challenges associated with transportation infrastructure, congestion, environmental impact, and overall quality of life. Bangalore, India, is no exception to these challenges.

Bangalore, often referred to as the "Silicon Valley of India," is a bustling metropolis known for its vibrant IT industry, educational institutions, and cultural diversity. With its booming economy and rapid urbanization, Bangalore faces significant transportation challenges, including traffic congestion, inadequate public transportation, and increasing commute times.

In this context, the analysis of commute data becomes invaluable for policymakers, urban planners, and residents seeking to understand and address these challenges effectively. The mini-project presented here focuses on analyzing commute data to and from a specific institution, RIT (Ramaiah Institute of Technology), situated in Bangalore.

The project aims to shed light on various aspects of commuting, including:

- Origin-Destination Patterns: Understanding where commuters are traveling from and how they are getting to RIT can provide insights into the city's transportation network and help identify areas with high demand for transportation infrastructure improvements.
- Mode Choice: Analyzing the modes of transportation chosen by commuters offers valuable information on the preferences, accessibility, and affordability of different transportation options. This insight can inform efforts to promote sustainable and efficient modes of transportation.
- Commute Time and Distance: Examining commute times and distances provides a comprehensive view of the commuting experience. Identifying areas with longer commute times or distances can guide efforts to optimize transportation routes, reduce congestion, and improve overall accessibility.
- Statistical Analysis: Calculating summary statistics such as average commute time, median distance traveled, and variance in commute

patterns offers quantitative insights into commuting trends and variability. These statistics can inform policy decisions, infrastructure investments, and urban planning strategies.

By analyzing and visualizing commute data, this mini-project aims to contribute to a deeper understanding of commuting patterns in Bangalore and provide valuable insights for stakeholders involved in transportation planning and management. Additionally, it serves as an educational exercise, demonstrating the application of Python programming and data analysis techniques to real-world transportation data.

In the subsequent sections of this report, we will provide a detailed overview of the code structure, libraries utilized, functions employed, and the insights gleaned from the analysis of commute data. Through this exploration, we aim to uncover meaningful patterns, trends, and implications for transportation planning and urban development in Bangalore.

Libraries and Functions used:

`csv`:
The `csv` library in Python provides functionality to work with Comma Separated Value (CSV) files. It allows reading from and writing to CSV files in a straightforward manner. With the `csv.writer()` function, you can create a writer object for writing data to CSV files. The `open()` function is used to open files for reading or writing, and `input()` is utilized to read input from the user via the command line.

Functions used:
`csv.writer()`: Creates a writer object for writing data to CSV files.
`open()`: Opens a file for reading or writing.
`input()`: Reads input from the user via the command line.

pandas:
`pandas` is a powerful data manipulation and analysis library in Python. It provides data structures and functions designed to make working with structured data easy and intuitive. The `pd.read_csv()` function is used to read data from CSV files into a pandas DataFrame, which is a two-dimensional labeled data structure. Functions like `groupby()` are used for grouping data based on specified criteria, while `size()` computes group sizes, and `reset_index()` resets the index of a DataFrame.

Functions used:
`pd.read_csv()`: Reads data from a CSV file into a pandas DataFrame.
`groupby()`: Groups DataFrame rows using a mapper or by a Series of columns.
`size()`: Computes the size of each group.
`reset_index()`: Resets the index of a DataFrame.

`tabulate`:

`tabulate` is a Python library that formats tabular data for printing in a visually appealing manner. It is particularly useful for presenting data in a human-readable format, especially when working with pandas DataFrames. The `tabulate()` function takes tabular data as input and formats it into a table format that can be easily printed to the console or exported to other formats like HTML or LaTeX.

Functions used:
`tabulate()`: Formats tabular data for printing in a visually appealing manner.

`numpy`:
`numpy` is a fundamental package for scientific computing with Python. It provides support for multidimensional arrays, along with a collection of mathematical functions to operate on these arrays efficiently. Functions like `np.mean()` calculate the arithmetic mean of array elements, `np.median()` computes the median, `np.var()` calculates the variance, `np.std()` computes the standard deviation, `np.max()` finds the maximum value, and `np.min()` finds the minimum value in an array.

Functions used:
`np.mean()`: Calculates the arithmetic mean of array elements.
`np.median()`: Computes the median of array elements.
`np.var()`: Calculates the variance of array elements.
`np.std()`: Computes the standard deviation of array elements.
`np.max()`: Finds the maximum value in an array.
`np.min()`: Finds the minimum value in an array.

`matplotlib.pyplot`:
`matplotlib.pyplot` is a plotting library for Python that provides a MATLAB-like interface for creating static, animated, and interactive visualizations. It is widely used for creating plots, charts, and graphs to visualize data. Functions like `plt.plot()` generate line plots, `plt.title()` sets the title of the plot, `plt.xlabel()` and `plt.ylabel()` set labels for the x-axis and y-axis

respectively, `plt.xticks()` sets the x-axis tick labels, and `plt.show()` displays the plot on the screen.

Functions used:

`plt.plot()`: Generates line plots.
`plt.title()`: Sets the title of the plot.
`plt.xlabel()`: Sets the label for the x-axis.
`plt.ylabel()`: Sets the label for the y-axis.
`plt.xticks()`: Sets the x-axis tick labels.
`plt.show()`: Displays the plot on the screen.

Code:

1. dataorg.py

```
import pandas as pd
from tabulate import tabulate

base_df = pd.read_csv('commute_data.csv')
location_counts =
base_df.groupby('LOC').size().reset_index(name='n')
mode_counts =
base_df.groupby('MOT').size().reset_index(name='n')
average_commute_time_byloc = base_df.groupby('LOC')['CT
(min)'].mean().reset_index()
average_commute_time_bymot = base_df.groupby('MOT')['CT
(min)'].mean().reset_index()
mode_usage_by_location = base_df.groupby(['LOC',
'MOT']).size().unstack(fill_value=0)

def main():
    print("Data used: ")
    print(tabulate(base_df, headers="keys",
tablefmt="psql"))
    print('\n')
    print("Number of commuters to RIT from various areas in
Bangalore: ")
```

```
    print(tabulate(location_counts, headers="keys",
tablefmt="psql"))
    print('\n')
    print("Number of commuters using various modes of
transportation on their commute to RIT: ")
    print(tabulate(base_df, headers="keys",
tablefmt="psql"))
    print('\n')
    print("Average Commute time by Location of commuter
origin: ")
    print(tabulate(base_df, headers="keys",
tablefmt="psql"))
    print('\n')
    print("Average Commute time by Mode of transport of
commute: ")
    print(tabulate(base_df, headers="keys",
tablefmt="psql"))
    print('\n')
    print("Mode of Transportation Distribution by Location:
")
    print(tabulate(base_df, headers="keys",
tablefmt="psql"))
    print('\n')

if __name__ == "__main__":
    main()
```

2. data_accept.py

```
import csv

VALID_LOC_ENTRIES = {
```

```
    'YEL': 'Yelahanka',
    'KRI': 'Krishnarajapuram',
    'BYA': 'Byatarayanapura',
    'YES': 'Yeshwantpur',
    'RAJ': 'Rajarajeshwarinagar',
    'DAS': 'Dasarahalli',
    'MAH': 'Mahalakshmi layout',
    'MAL': 'Malleshwaram',
    'HEB': 'Hebbal',
    'PUL': 'Pulakeshinagar',
    'SAR': 'Sarvagnagar',
    'CVR': 'CV Raman Nagar',
    'SHI': 'Shivajinagar',
    'SHA': 'Shantinagar',
    'GAN': 'Gandhinagar',
    'GOV': 'Govindrajnagar',
    'VIJ': 'Vijaynagar',
    'CHA': 'Chamrajpet',
    'CHI': 'Chickpet',
    'BAS': 'Basavanagudi',
    'PAD': 'Padmanabhanagar',
    'BTM': 'BTM Layout',
    'JAY': 'Jayanagar',
    'MAH': 'Mahadevapura',
    'BOM': 'Bommanahalli',
    'BNS': 'Bangalore South',
    'ANE': 'Anekal'
}

VALID_MOT_ENTRIES = {
    'C': 'Car',
    'B': 'Bus',
    'T': 'Two-wheeler',
    'W': 'Walk',
```

```
    'M': 'Miscellaneous'
}

def main():
    with open('commute_data.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['LOC', 'MOT', 'CT (min)', 'DIST
(km)'])

    while True:
        loc = input("Enter location (LOC - Valid entries:
{}): ".format(', '.join(VALID_LOC_ENTRIES.keys())))
        while loc.upper() not in VALID_LOC_ENTRIES:
            print("Invalid location entry. Please choose
from the valid entries.")
            loc = input("Enter location (LOC - Valid
entries: {}): ".format(', '.join(VALID_LOC_ENTRIES.keys())))

        mot = input("Enter mode of transportation (MOT -
Valid entries: {}): ".format(',
'.join(VALID_MOT_ENTRIES.keys())))
        while mot.upper() not in VALID_MOT_ENTRIES:
            print("Invalid mode of transportation entry.
Please choose from the valid entries.")
            mot = input("Enter mode of transportation (MOT
- Valid entries: {}): ".format(',
'.join(VALID_MOT_ENTRIES.keys())))

        commute_time = input("Enter commute time in
minutes (CT): ")
        distance = input("Enter distance between college
and point of origin in kilometers (DIS): ")
```

```
        writer.writerow([VALID_LOC_ENTRIES[loc.upper()],
VALID_MOT_ENTRIES[mot.upper()], commute_time, distance])

        add_more = input("Do you want to add more commute
data? (yes/no): ").lower()
        if add_more != 'yes' and add_more != 'y':
            break

if __name__ == "__main__":
    main()
```

3. visualizer.py

```
import matplotlib.pyplot as plt
from dataorg import *

location_counts.plot(x = 'LOC', y = 'n', kind = "bar", color
= "lime", width = 0.8)
plt.title('Number of commuters to RIT from various areas in
Bangalore')
plt.xlabel('Locations -->')
plt.ylabel('Number of commuters -->')
plt.xticks(rotation=45)
plt.show()

mode_counts.plot(x = 'MOT', y = 'n', kind = "bar", color =
"skyblue", width = 0.8)
plt.title('Number of commuters using various modes of
transportation on their commute to RIT')
plt.xlabel('Mode of transportation -->')
plt.ylabel('Number of commuters -->')
plt.xticks(rotation=45)
plt.show()
```

```
average_commute_time_byloc.plot(x = 'LOC', y = 'CT (min)',
kind = 'bar', color = "crimson", width = 0.8)
plt.title('Average Commute time by Location of commuter
origin')
plt.xlabel('Locations -->')
plt.ylabel('Average Commute time -->')
plt.xticks(rotation=45)
plt.show()

average_commute_time_bymot.plot(x = 'MOT', y = 'CT (min)',
kind = 'bar', color = "orange", width = 0.8)
plt.title('Average Commute time by Mode of transport of
commute')
plt.xlabel('Modes of transport -->')
plt.ylabel('Average Commute time -->')
plt.xticks(rotation=45)
plt.show()

mode_usage_by_location.plot(kind='bar', stacked=True,
figsize=(10, 6))
plt.title('Mode of Transportation Distribution by Location')
plt.xlabel('Location')
plt.ylabel('Number of Commuters')
plt.xticks(rotation=45)
plt.show()
```

4. summary.py

```
from dataorg import *
import numpy as np

def main():
    print("Commute time statistics: ")
```

```
    commute_time_mean = round(np.mean(base_df['CT
(min)']),2)
    print(f"Average Commute time = {commute_time_mean}
min")

    commute_time_median = round(np.median(base_df['CT
(min)']),2)
    print(f"Median of Commute time = {commute_time_median}
min")

    commute_time_var = round(np.var(base_df['CT (min)']),2)
    print(f"Variance of Commute time = {commute_time_var}
min")

    commute_time_std = round(np.std(base_df['CT (min)']),2)
    print(f"Standard deviation of Commute time =
{commute_time_std} min")

    commute_time_range = round(np.max(base_df['CT (min)'])
- np.min(base_df['CT (min)']),2)
    print(f"Range of Commute time = {commute_time_range}
min")

    print('\n')

    print("Distance statistics: ")
    dist_mean = round(np.mean(base_df['DIST (km)']),2)
    print(f"Average Distance travelled = {dist_mean} km")

    dist_median = round(np.median(base_df['DIST (km)']),2)
    print(f"Median of Distance travelled = {dist_median}
km")

    dist_var = round(np.var(base_df['DIST (km)']),2)
```

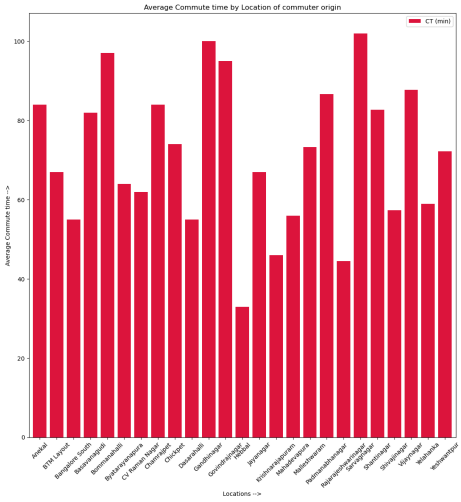
```
    print(f"Variance of Distance travelled = {dist_var}
km")

    dist_std = round(np.std(base_df['DIST (km)']),2)
    print(f"Standard Deviation of Distance travelled =
{dist_std} km")

    dist_range = round(np.max(base_df['DIST (km)']) -
np.min(base_df['DIST (km)']),2)
    print(f"Range of Distance travelled = {dist_range} km")

if __name__ == "__main__":
    main()
```

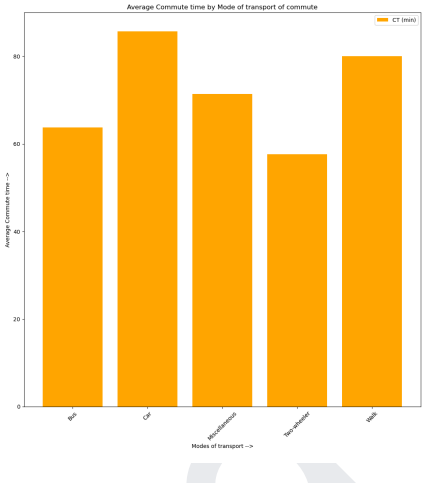
Output:

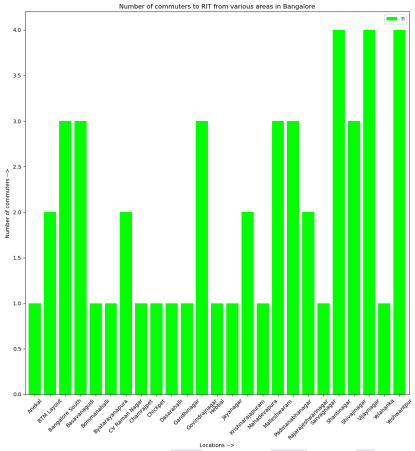
github repository link:
https://github.com/themohitnair/commute_mitra

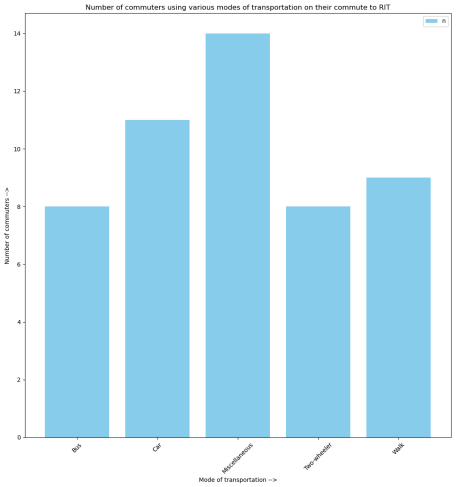
Text output file (too big to include here):
https://github.com/themohitnair/commute_mitra/blob/main/output.txt

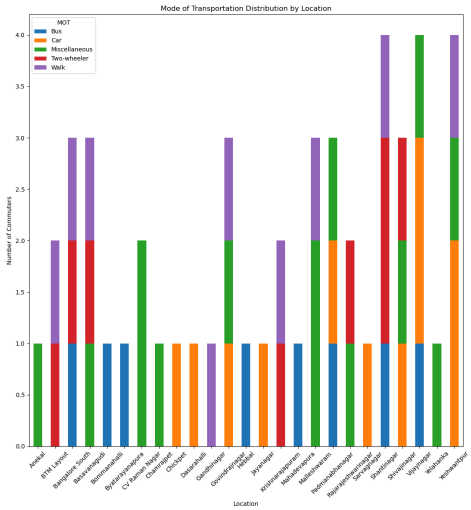
Image output directory (matplotlib plots):
https://github.com/themohitnair/commute_mitra/tree/main/output-pictures (also added in the next page)


Average Commute time by Location of commuter origin
CT (min)
Average Commute time -->
Locations -->


Average Commute time by Mode of transport of commute
CT (min)
Average Commute time -->
Bus
Car
Miscellaneous
Two-wheeler
Walk
Modes of transport -->


Number of commuters to RIT from various areas in Bangalore
n
Number of commuters -->
Locations -->


Number of commuters using various modes of transportation on their commute to RIT
n
Number of commuters -->
Bus
Car
Miscellaneous
Two-wheeler
Walk
Mode of transportation -->


Mode of Transportation Distribution by Location
MOT
Bus
Car
Miscellaneous
Two-wheeler
Walk
Number of Commuters
Location