CS 573 Process Book

Dong Tang, Anthony Chen, Songling Li, Jinqin Xiong

I. Overview and Motivation:

Overview and Motivation: Provide an overview of the project goals and the motivation for it. Consider that this will be read by people who did not see your project proposal.

- Our project aims to incorporate features such as clusters and dynamic movements. Given our team members' background in backend development with geographic data, we decided to focus on projects related to geography.
- After establishing the general direction, we set up to find relevant datasets. Initially, we discovered the New York City taxi dataset, which was comprehensive, along with some related weather data. After that, we began to concentrate on map development. We found out that with the right dataset, we can obtain information on passenger pick-up and drop-off locations and times, and generate visuals not only on taxi pick-up & drop-off clusters and paths, but also on travel distances, speeds, commutes between clusters and broken line analysis during the day.

New York Taxi Data:
https://learn.microsoft.com/zh-cn/azure/open-datasets/dataset-taxi-yellow?tabs=azureml-opendatasets

Weather Data:
https://learn.microsoft.com/en-us/python/api/azureml-opendatasets/azureml.opendatasets.noaaisdweather?view=azure-ml-py

Shen Zhen Taxi Data
https://www.nbsdc.cn/general/dataLinks/16666.11.nbsdc.y3xucigm

II. Related work:

Related Work: Anything that inspired you, such as a paper, a website, visualizations we discussed in class, etc.

1. Bubbles Charts Visualization

- In my project, I was particularly inspired by the article titled "How to Create Storytelling Moving Bubbles Charts in d3js with Python," authored by Erdogan Taskesen and published on Towards Data Science. This article thoroughly explores the use of moving bubble charts for effective storytelling and data visualization across time and states. The MovingBubble chart, as described, is an impactful tool for illustrating the distribution and dynamics of individual

items, leveraging the D3Blocks library built on d3 JavaScript and configurable via Python.

- The article emphasizes the utility of moving bubble charts in visualizing complex data sets interactively, where elements such as bubbles move across a predefined timeline to represent different states. Such visualizations offer a novel approach to understanding data through animated transitions, making it easier to perceive changes over time and patterns within the data. Taskesen explains the technical process involved in creating these charts, including data preparation, parameter settings, and the final rendering in a web browser, which aligns with the goals of my project to harness advanced visualization techniques for more dynamic data interpretation.
- The integration of Python for data preprocessing and d3 JavaScript for visualization provides a flexible, powerful platform for developing interactive charts that are both informative and visually engaging. This methodology aligns with my project's aim to employ sophisticated data visualization tools to enhance the presentation and understanding of complex datasets.

For more details on this methodology and its applications, the full article can be accessed at [How to Create Storytelling Moving Bubbles Charts in d3js with Python | by Erdogan Taskesen | Towards Data Science]

2. Geospatial Data Visualization

- The endeavor to enhance geospatial data visualization led me to develop a 3D visualization of taxi boarding points across Shenzhen. This visualization not only captures the geographical distribution of these points but also integrates the dimension of average travel distances, elevating the depth of analysis and insight provided.

More details about this technique can be found on
https://plotly.com/python/3d-charts/

III. Questions

Questions: What questions are you trying to answer? How did these questions evolve over the course of the project? What new questions did you consider in the course of your analysis?

1. What questions are you trying to answer?

1. How do taxi pick-up and drop-off patterns vary throughout the day?
2. What are the most frequent pick-up and drop-off locations?
3. How does traffic flow change during different times of the day?

2. As the project progressed, our questions evolved to incorporate more detailed inquiries, including:

1. Can we identify peak traffic times and locations to optimize taxi dispatch and availability?
2. How do external factors like weather impact taxi travel patterns?

3. New questions considered during our analysis included:

1. Can we predict future taxi demand based on historical data?
2. How might changes in urban infrastructure or public policy impact taxi usage patterns?

IV. Data

Data: Source, scraping method, cleanup, etc.

- After consulting with WPI Data Science Professor Yanhua Li, we obtained a dataset containing records of taxi movements in Shenzhen from a large .pkl file, which was then loaded into a pandas DataFrame for easier analysis. This dataset includes details like plate number, coordinates, timestamp, speed, and acceleration, documenting the taxis' statuses at specific times, indicating whether they were carrying passengers or idle. By analyzing these data fields, particularly longitude, latitude, and timestamp, we computed each taxi's speed and acceleration, crucial for understanding taxi movement dynamics, driving patterns, and identifying anomalies. Subsequently, we processed this data to extract essential information and saved it in a .json file for further analysis.
- Furthermore, We conducted DataFrame expansion to reorganize nested list structures within the dataset into a more manageable form, which is particularly useful when dealing with complex datasets involving geographical or temporal data. Subsequently, We performed timestamp data transformation to convert it into a more convenient datetime format and extracted various

components of the date such as year, month, and day. This step facilitated time-based aggregation and analysis.

- In terms of data visualization, We utilized geopandas and matplotlib tools for spatial data operations and plotting. By mapping taxi boarding and alighting points on a map of Shenzhen, We provided visual representations of taxi activities throughout the city. Additionally, during the clustering analysis phase, We employed the K-means algorithm to cluster spatial coordinates and identify hotspots of taxi activity, which holds significance for urban planning and optimizing taxi dispatch. Moreover, We processed boarding and alighting data into JSON format, enabling easy data interchange across different platforms or web applications.

Data Processing File(.ipynb):
https://colab.research.google.com/drive/1B36tKcVbZ3m5pEmKQCe4eN6Seo1JIn77?usp=sharing

Partial screenshots of the file

```
[ ] import pickle
    from IPython.display import display, Javascript
    import pandas as pd


    # 路径
    file_path = '/content/drive/My Drive/504Project4/data/train_data.pkl'


    with open(file_path, 'rb') as f:
            data = pickle.load(f)

    # 将数据转换为 DataFrame
    train_df = pd.DataFrame(data)
```

```
[ ] #特征提取，创建速度列以及加速度列
    #Feature extraction, creating speed column and acceleration column

    import numpy as np
    import pandas as pd

    def calculate_speed_and_acceleration(df):
            # 提取经度和纬度列
            longitude = df.iloc[:, 1]
            latitude = df.iloc[:, 2]

            # 提取时间列
            time_seconds = df.iloc[:, 3]

            # 计算两个轨迹点之间的距离
            distances = np.sqrt((longitude.diff() ** 2) + (latitude.diff() ** 2))

            # 计算速度
            df['speed'] = distances / time_seconds.diff()

            # 计算加速度
            df['acceleration'] = df['speed'].diff() / time_seconds.diff()

            return df
```

```
[ ] #改变复杂数据集为扁平化
```

:

```
#dataframe Flattening the complex dataset into a dataframe.

#flattening the test dataset.
#测试集扁平化

import pandas as pd


columns = ['Plate', 'Longitude', 'Latitude', 'Seconds Since Midnight', 'Status', 'Time']


expanded_frames = []

# 迭代每个列
for col in train_df.columns:
        # 每个单元格是一个轨迹的列表
        col_data = []
        for cell in train_df[col].dropna():    # 确保非空
                if isinstance(cell, list):

                        if len(cell) > 0 and isinstance(cell[0], list):
                                # 将列表转换为DataFrame
                                temp_df = pd.DataFrame(cell, columns=columns)
                                col_data.append(temp_df)
                        else:
                                print("Ignored cell due to incorrect format or empty list:", cell)
                else:
                        print("Ignored cell due to not being list:", cell)

        # 如果列数据非空，合并为一个DataFrame
        if col_data:
                col_df = pd.concat(col_data, ignore_index=True)
                expanded_frames.append(col_df)
        else:
                print("No valid data found in column:", col)

# 检查是否有DataFrame可以合并
if expanded_frames:
        # 合并所有列的数据为一个大的DataFrame
        expanded_df = pd.concat(expanded_frames, ignore_index=True)
        # 展示结果
        print(expanded_df.head())
else:
        print("No frames to concatenate. Expanded_frames is empty.")

#name
expand_train_df = expanded_df.copy()
```

V. Exploratory Data Analysis

Exploratory Data Analysis: What visualizations did you use to initially look at your data? What insights did you gain? How did these insights inform your design?

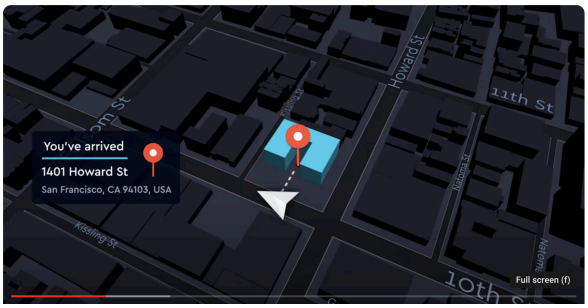
Initially, we searched through various online map display formats, drawing inspiration from existing designs and integrating them with our dataset to create our own web-based visualization. What we have searched as below links

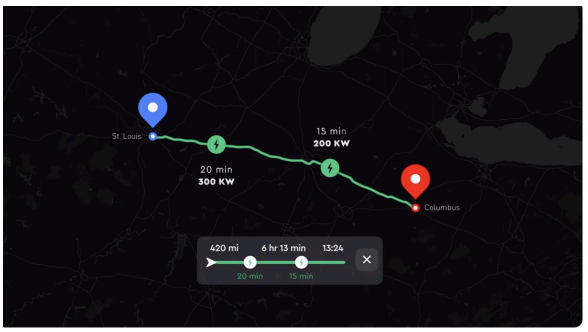
1. Map Visualization Link:

https://www.youtube.com/watch?v=D57xNfU73as

2. Mapbox Package:

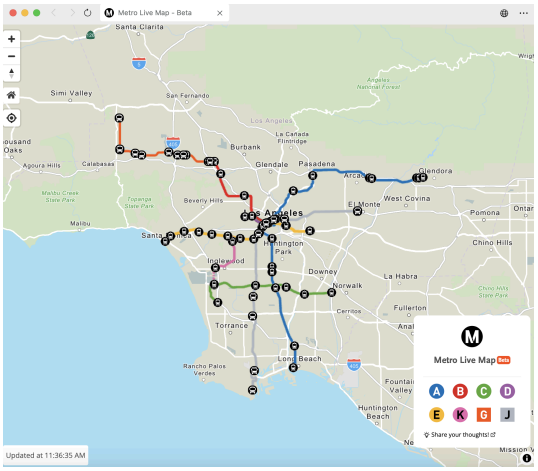
https://www.youtube.com/watch?v=TpsaBnfQVuM





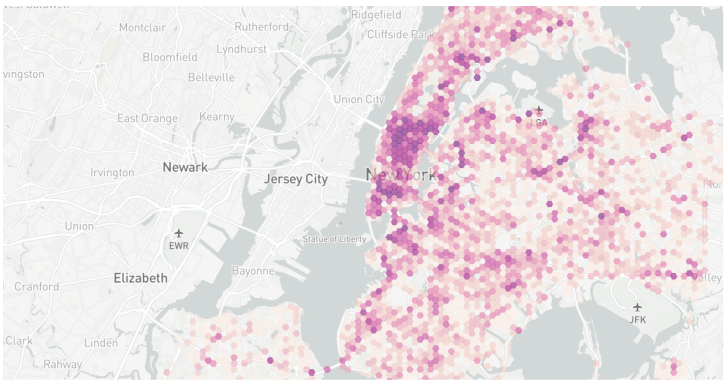
3. Real-time Map Visualization:

https://lacmta.github.io/realtime-map/



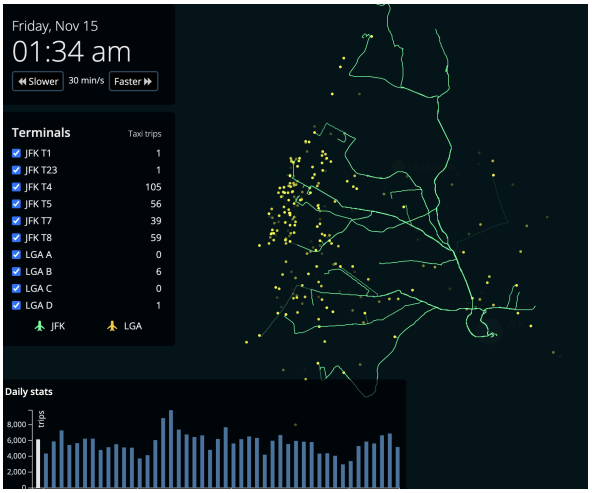
4. Mapboxgl Link:

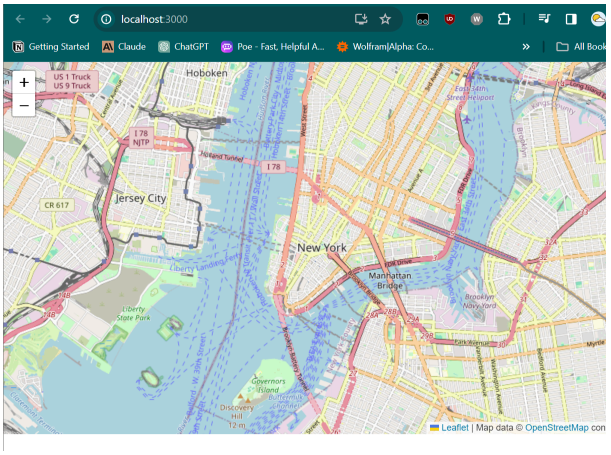
https://observablehq.com/@clhenrick/mapboxgl-hexbin-map



5. NYC Taxi Visualization：

https://taxi.imagework.com/





VI. Design Evolution

Design Evolution: What are the different visualizations you considered? Justify the design decisions you made using the perceptual and design principles you learned in the course. Did you deviate from your proposal?

1. What are the different visualizations you considered?

- We explored a range of visualization techniques before finalizing our design choices. Each decision was guided by perceptual and design principles aimed at enhancing user understanding and interaction with the data.
- Line graphs were considered to show trends over time, such as the fluctuation of taxi demand. Also, inspired by the moving bubbles concept we discussed in class and found in related works, we implemented a moving bubbles chart to

dynamically represent the flow of taxis over time, making the data more relatable and easier to understand in a temporal context.

- 3D visualization technique is an advanced tool. By representing average travel distances as varying heights on a geographical map, this visualization technique transcends traditional 2D mapping, offering a more dynamic and layered perspective of the data.

2. Justify the design decisions you made using the perceptual and design principles you learned in the course.

- According to perceptual principles, interactive visualizations engage users more effectively by allowing them to explore the data in ways that static images cannot. This approach leverages the user's active participation to enhance memory retention and understanding.
- moving bubbles was a decision driven by the need to depict dynamic changes within the data set over time. This method aligns with the principle of temporal representation, which is crucial for understanding patterns in datasets where time is a significant variable.

3. Did you deviate from your proposal?

- Due to the complexity and technical challenges associated with merging these datasets effectively, we shifted our focus solely to taxi data visualization, and changed the place from New York to Shen Zhen. This pivot was necessary to maintain the project's feasibility and ensure that we could deliver a robust and functional visualization tool within the project's timeline.

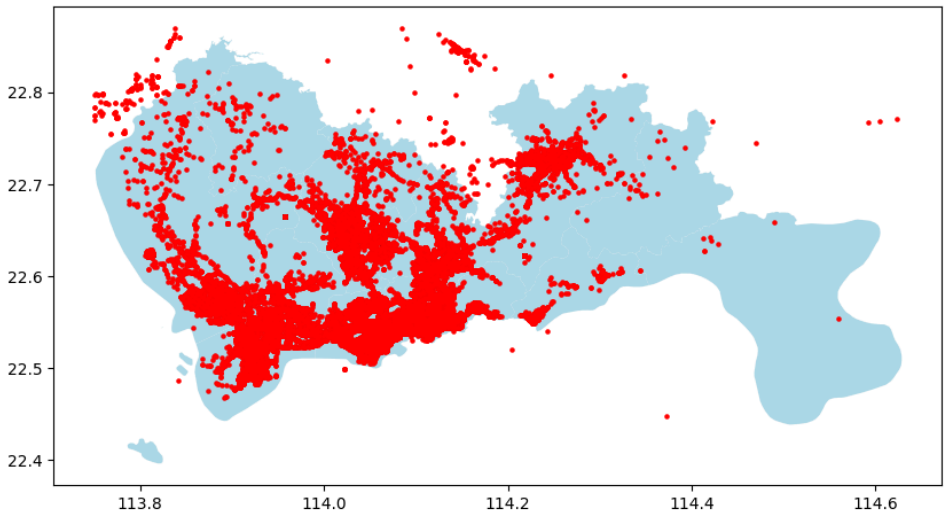
VII. Implementation

Implementation: Describe the intent and functionality of the interactive visualizations you implemented. Provide clear and well-referenced images showing the key design and interaction elements.

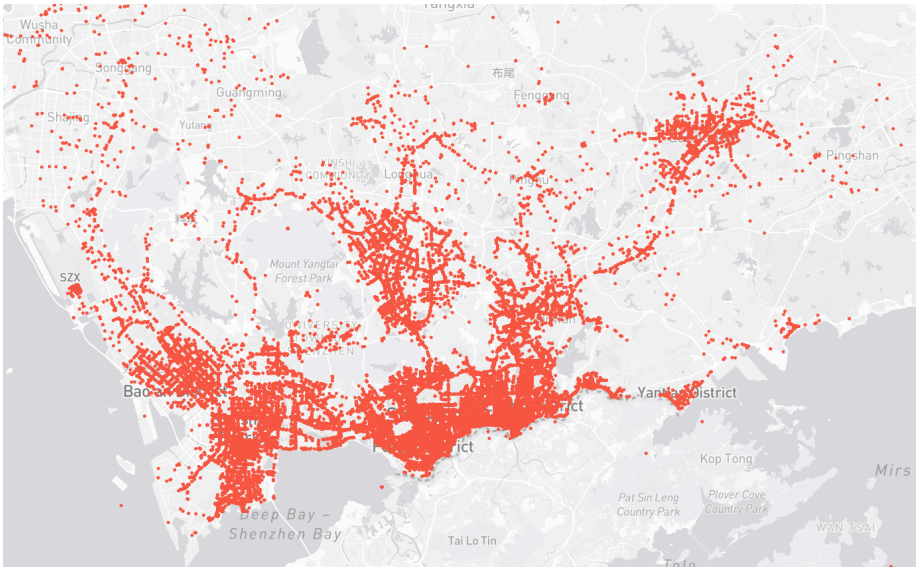
Note: For the pick-up & drop-off points, clusters, and path visualization, we got the idea from another place and made a different implementation in our own way. For the bubble, 3D and broken-line visualization, we come up with the idea and implemented them by ourselves.

1. A simulation of the pick-up & drop-off points on the map

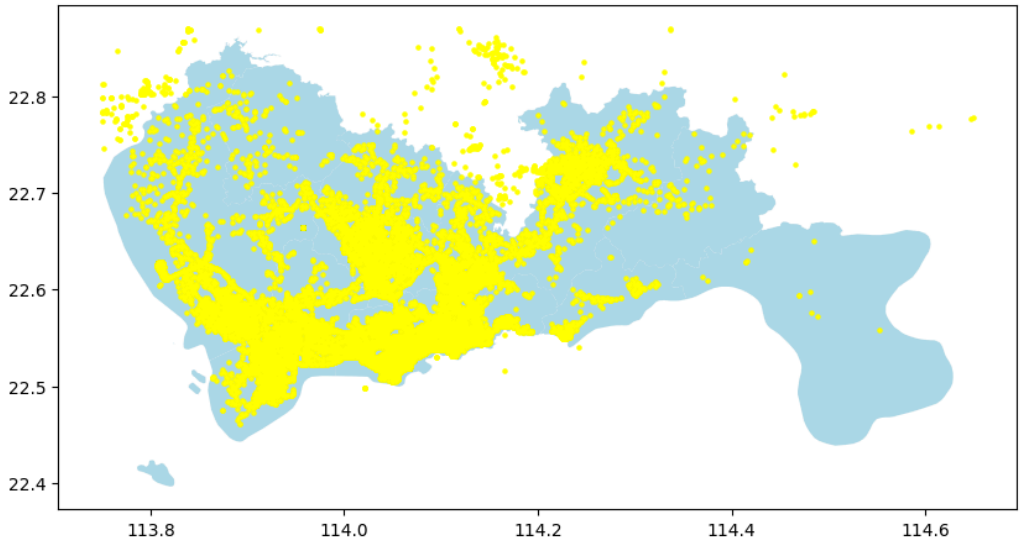
```
import geopandas as gpd
import matplotlib.pyplot as plt
import json
from shapely.geometry import Point

# 加载深圳市的地理数据
shenzhen = gpd.read_file('/content/drive/MyDrive/DV Final/shenzhenshi.json')

# 加载坐标点数据
with open('/content/drive/MyDrive/DV Final/boarding_json_data.json', 'r') as f:
    boarding_data = json.load(f)

# 根据提供的JSON格式，提取坐标点并创建GeoDataFrame
points = [Point(item['coordinates'][0], item['coordinates'][1]) for item in boarding_data]
points_gdf = gpd.GeoDataFrame(geometry=points)

# 绘制地图
fig, ax = plt.subplots(figsize=(10, 10))
shenzhen.plot(ax=ax, color='lightblue')    # 绘制深圳市的地区界线
points_gdf.plot(ax=ax, color='red', markersize=5)    # 在地图上标出坐标点，并将标记设置为更小
plt.show()
```



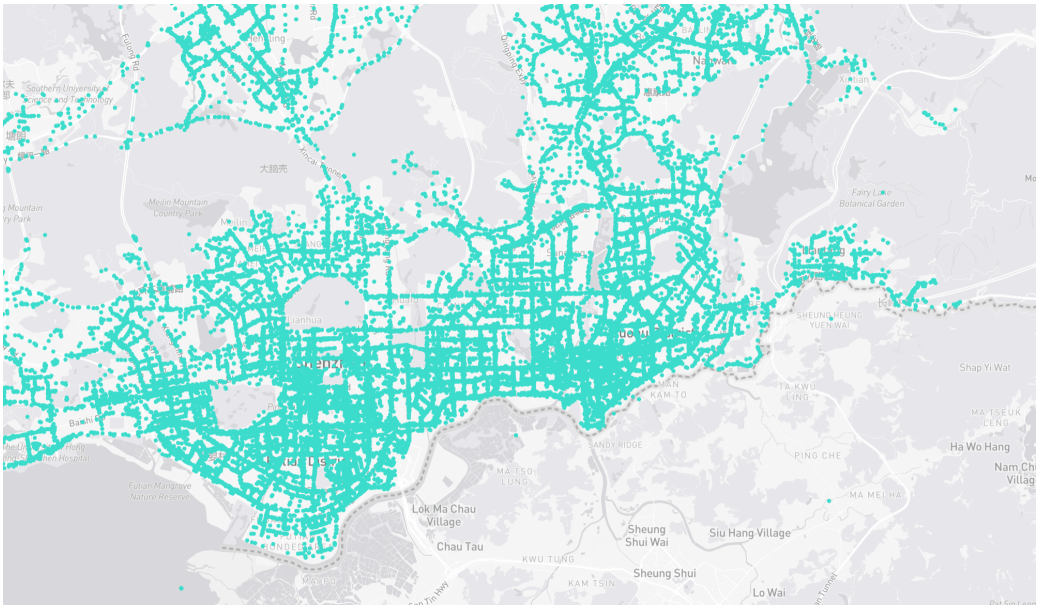
another visualization on the actual map



```
[ ] import geopandas as gpd
    import matplotlib.pyplot as plt
    import json
    from shapely.geometry import Point

    # 加载深圳市的地理数据
    shenzhen = gpd.read_file('/content/drive/MyDrive/DV Final/shenzhenshi.json')

    # 加载坐标点数据
    with open('/content/drive/MyDrive/DV Final/alighting_json_data.json', 'r') as f:
            alighting_data = json.load(f)

    # 根据提供的JSON格式，提取坐标点并创建GeoDataFrame
    points = [Point(item['coordinates'][0], item['coordinates'][1]) for item in alighting_data]
    points_gdf = gpd.GeoDataFrame(geometry=points)

    # 绘制地图
    fig, ax = plt.subplots(figsize=(10, 10))
    shenzhen.plot(ax=ax, color='lightblue')    # 绘制深圳市的地区界线
    points_gdf.plot(ax=ax, color='yellow', markersize=5)    # 在地图上标出坐标点，并将标记设置为更小
    plt.show()
```

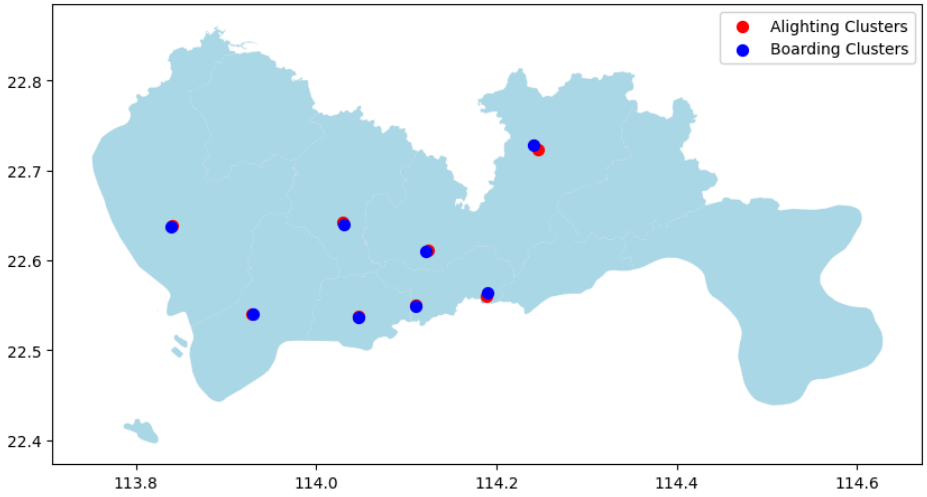


another visualization on the actual map:

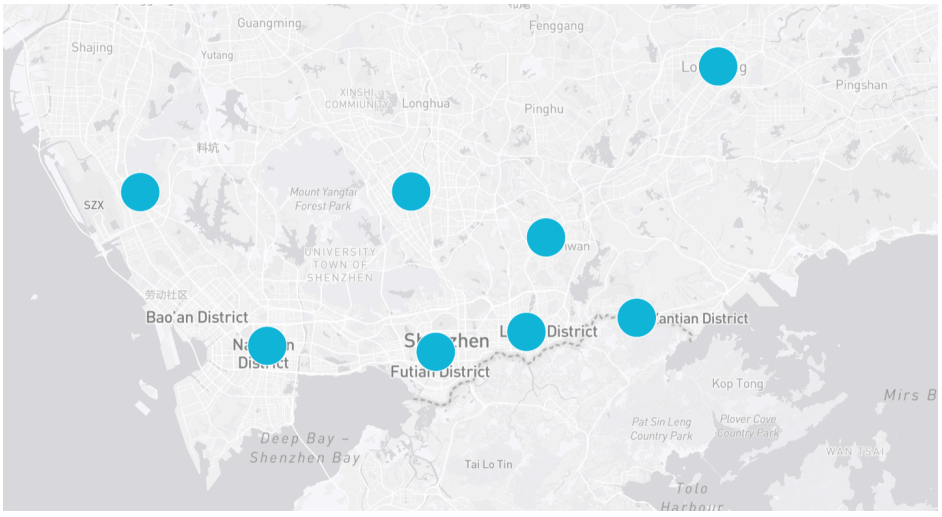


2. A simulation of the pick-up & drop-off clusters on the map

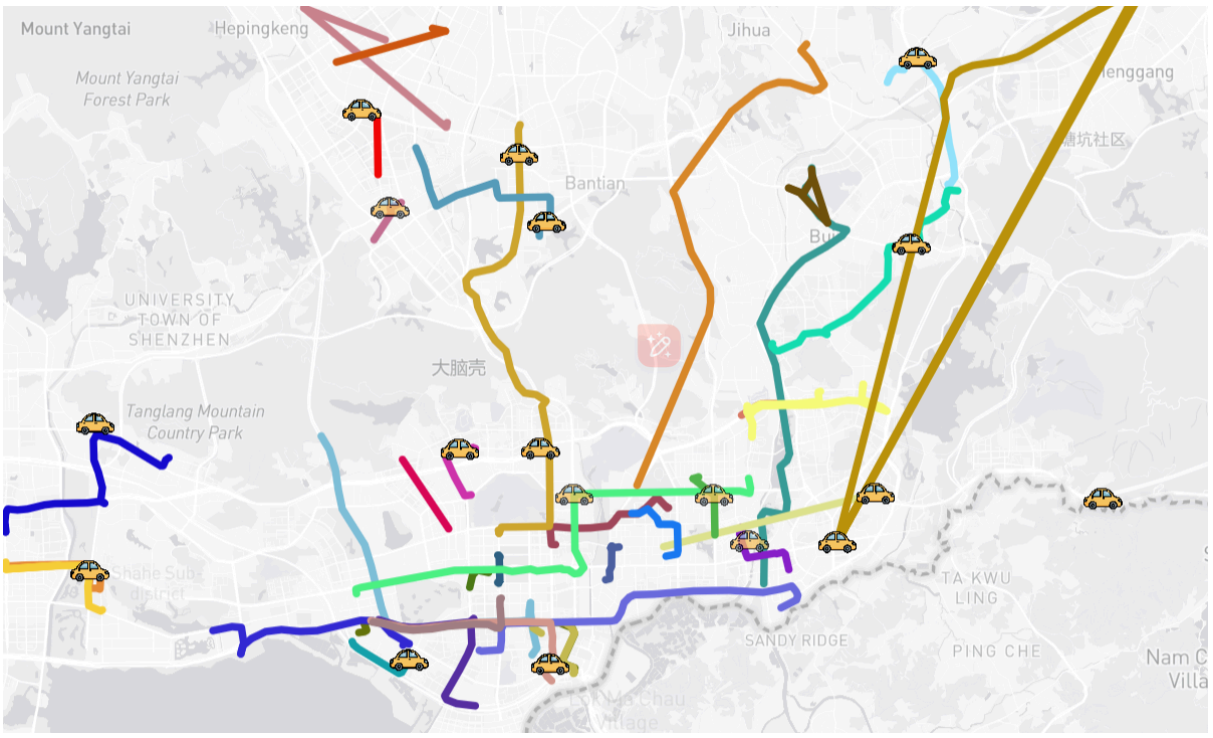
```
[ ] import geopandas as gpd
    import matplotlib.pyplot as plt
    import json
    from shapely.geometry import Point

    # 加载深圳市的地理数据
    shenzhen = gpd.read_file('/content/drive/MyDrive/DV Final/shenzhenshi.json')

    # 加载 alighting_cluster_json_data.json 文件中的聚类中心坐标
    with open('/content/drive/MyDrive/DV Final/alighting_cluster_json_data.json', 'r') as f:
        alighting_centers = json.load(f)

    # 加载 boarding_cluster_json_data.json 文件中的聚类中心坐标
    with open('/content/drive/MyDrive/DV Final/boarding_cluster_json_data.json', 'r') as f:
        boarding_centers = json.load(f)

    # 创建 GeoDataFrame 存储聚类中心坐标
    alighting_points = [Point(center[0], center[1]) for center in alighting_centers]
    alighting_gdf = gpd.GeoDataFrame(geometry=alighting_points)

    boarding_points = [Point(center[0], center[1]) for center in boarding_centers]
    boarding_gdf = gpd.GeoDataFrame(geometry=boarding_points)

    # 绘制地图
    fig, ax = plt.subplots(figsize=(10, 10))
    shenzhen.plot(ax=ax, color='lightblue')  # 绘制深圳市的地区界线
    alighting_gdf.plot(ax=ax, color='red', markersize=50, label='Alighting Clusters')  # 在地图上标出 alighting 聚类中心
    boarding_gdf.plot(ax=ax, color='blue', markersize=50, label='Boarding Clusters')  # 在地图上标出 boarding 聚类中心
    plt.legend()  # 显示图例
    plt.show()
```



another one but on the actual map:



another visualization on the Path



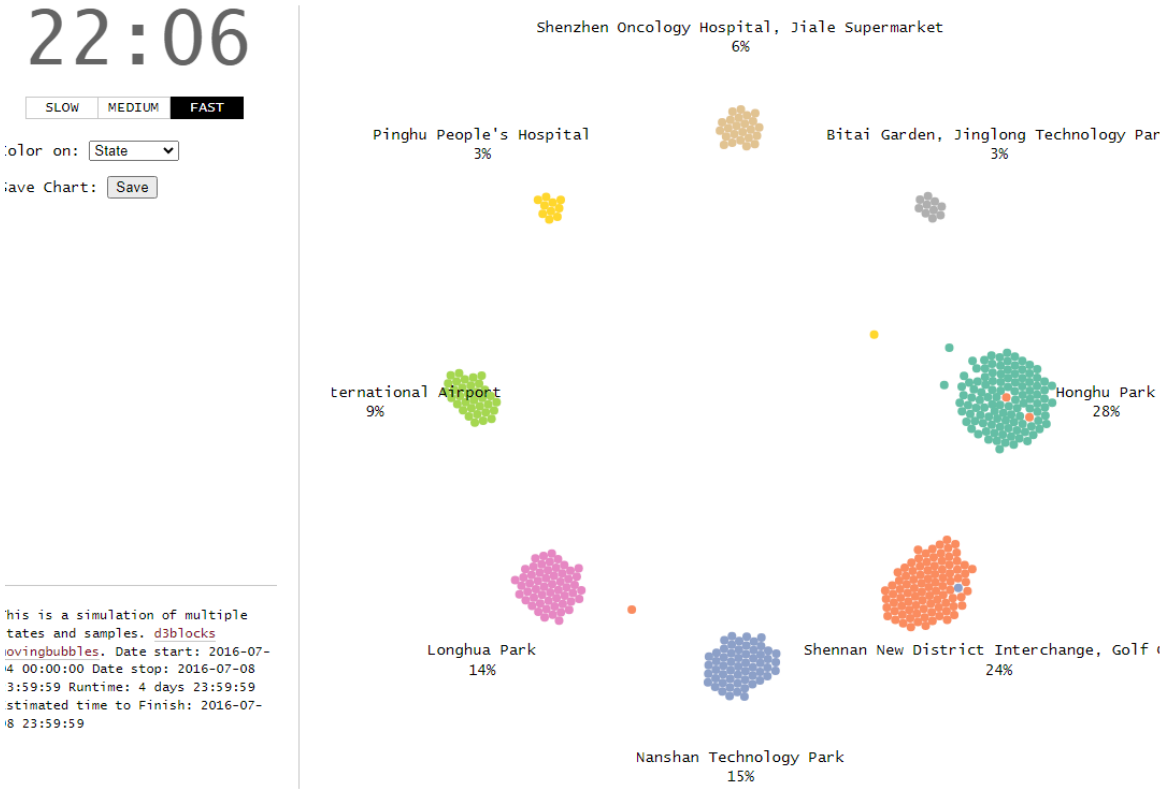
Besides the maps above, we have also conducted another three kinds of visualizations of the data.

3. **Bubble chart dynamic visualization**

We use the d3blocks library to create a dynamic bubble chart defining colors for each state and setting the size of the bubbles. Configure chart behavior, such as specifying color change methods, time units, and animation speeds. After generating the dynamic bubble chart and creating an HIMl file, we import the HiMl into our GitHub page and then integrate this page into the front end of a React application. Then, we can visually observe the changes in the position of taxis between different aggregated locations over a day, allowing for a deeper understanding of the dynamic patterns in the data.

This advanced visualization method effectively depicted the temporal dynamics of the data, and visualized taxi activity interactively, laying a solid foundation for further analysis and insights. The entire data processing and visualization pipeline not only presented the features of taxi movement data clearly but also established a robust groundwork for in-depth data analysis and insights.

```python
from d3blocks import D3Blocks

# 初始化D3Blocks库
d3 = D3Blocks()

# 设置每个样本的气泡大小，假设所有气泡大小相同
sizes = {str(i): 10 for i in taxi_bubble_df['sample_id'].unique()}    # 样本数量可能很多，设置较小的默认大小

# 为每个地点生成唯一颜色
colors = {state: f"#{hex(hash(state))[2:8]}" for state in taxi_bubble_df['state'].unique()}

# 创建动态气泡图
d3.movingbubbles(
        taxi_bubble_df,
        datetime='datetime',
        state='state',
        sample_id='sample_id',
        size=sizes,
        color=colors,
        color_method='state',     # 根据状态改变颜色
        timedelta='seconds',     # 根据数据的时间单位选择合适的timedelta
        speed={"slow": 1000, "medium": 100, "fast": 10},
        filepath=r'/content/drive/MyDrive/DV Final/movingbubbles.html',
        cmap='Set2',
        standardize='samplewise',
)

print("动态气泡图已生成，请查看指定的HTML文件。")
```



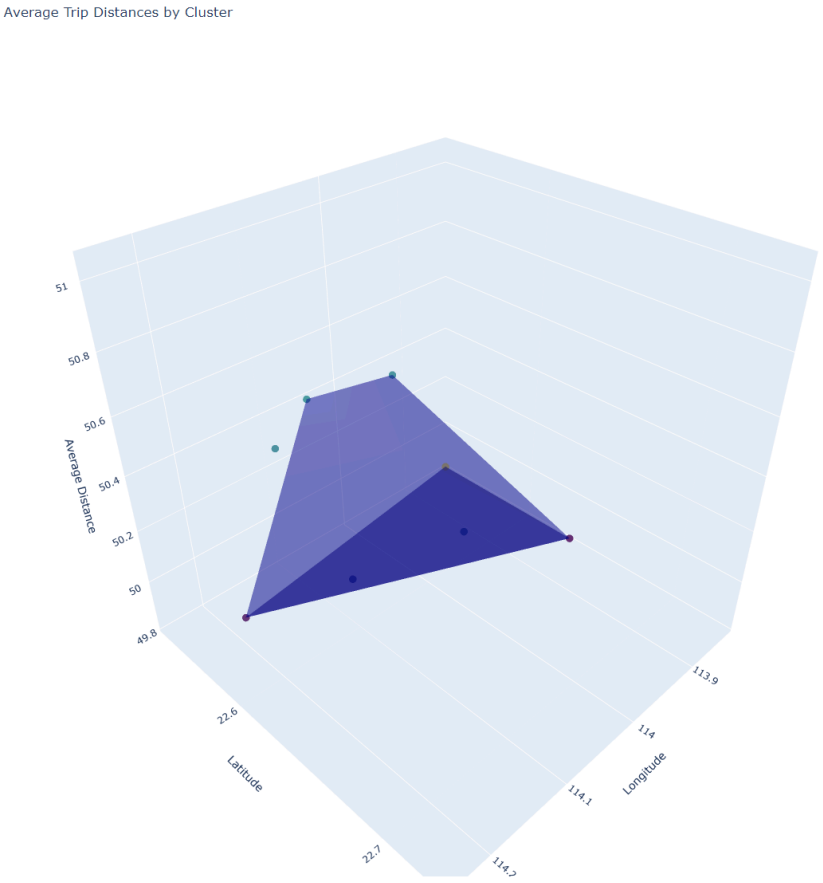
HTML for Bubble chart: https://croon233.github.io/DVFinal/

4. 3D visualization

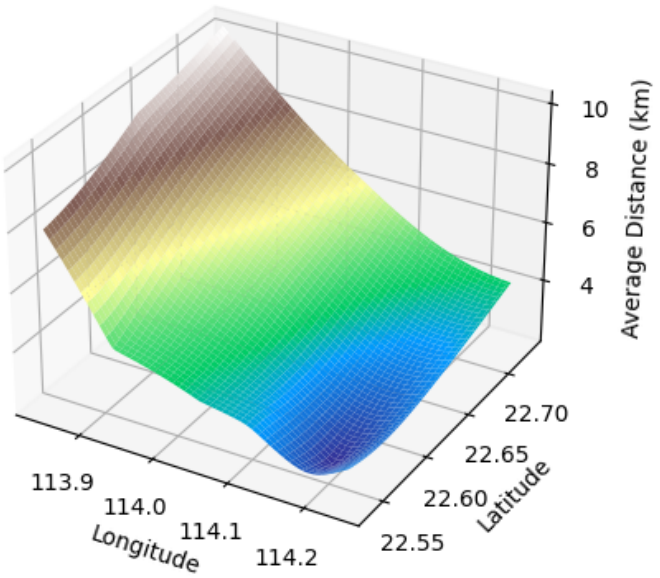
We conducted 3D visualization of taxi trip data by first preprocessing the data in Python, where we assigned each boarding point to its nearest cluster and calculated the average distances for trips originating from each cluster. Using Python libraries such as numpy, pandas, and scipy.spatial, we efficiently managed the data calculations. This allowed us to utilize matplotlib and mpl_toolkits.mplot3d to render a 3D plot. The plot visually represents clusters with their respective average distances along the z-axis, depicting the spatial relationships between different taxi starting points.

Further enhancing the 3D visualization, we incorporated a convex shape to illustrate the gradient process, visualizing how trip distances vary with proximity to the city center. As illustrated, starting points closer to the city center correlate with shorter trip distances, whereas those farther away tend to have longer distances. We employed Plotly's Scatter3d to create an interactive 3D scatter plot. We configured the visualization to display in the browser and also saved it as an HTML file to Google Drive for easy sharing. This approach leverages both data manipulation and visualization techniques, enhancing the understanding of urban traffic flow and taxi operation dynamics.

```
import pandas as pd
import plotly.graph_objects as go
from scipy.spatial import ConvexHull
from google.colab import drive

# 连接到 Google Drive
drive.mount('/content/drive')

import numpy as np

# Suppose `cluster_centers` is a list of [longitude, latitude] for each cluster center
# and `boarding_coordinates` is a DataFrame containing columns ['Longitude', 'Latitude']

def assign_to_nearest_cluster(boarding_coordinates, cluster_centers):
    # Convert cluster_centers into a numpy array for efficient computation
    cluster_centers = np.array(cluster_centers)
    assignments = []

    for index, row in boarding_coordinates.iterrows():
        # Calculate the distance to each cluster center
        distances = np.sqrt((cluster_centers[:, 0] - row['Longitude'])**2 + (cluster_centers[:, 1] - row['Latitude'])**2)
        # Find the index of the minimum distance
        nearest_cluster = np.argmin(distances)
        assignments.append(nearest_cluster)

    boarding_coordinates['Cluster'] = assignments
    return boarding_coordinates

boarding_coordinates = assign_to_nearest_cluster(boarding_coordinates, cluster_centers)

# Adding a dummy 'Distance' column to simulate trip distances
boarding_coordinates['Distance'] = np.random.rand(len(boarding_coordinates)) * 100  # Random distances between 0 and 100

# Calculate the average distance per cluster
average_distances = boarding_coordinates.groupby('Cluster')['Distance'].mean()

import plotly.graph_objects as go

fig = go.Figure(data=[go.Scatter3d(
    x=cluster_centers[:, 0],
    y=cluster_centers[:, 1],
    z=average_distances,
    mode='markers+lines',
    marker=dict(
        size=12,
        color=average_distances,    # Color the points by average distance
        colorscale='Viridis',         # Choose a color scale
        opacity=0.8
    ),
    line=dict(
        color='darkblue',
        width=2
    )
)])

fig.update_layout(scene=dict(
                                xaxis_title='Longitude',
                                yaxis_title='Latitude',
                                zaxis_title='Average Distance'),
                            title='Average Trip Distances by Cluster')
fig.show()

# 保存 HTML 文件到 Google Drive
fig.write_html('/content/drive/MyDrive/Colab Notebooks/DV Final/cluster_distances.html', auto_open=True)
```

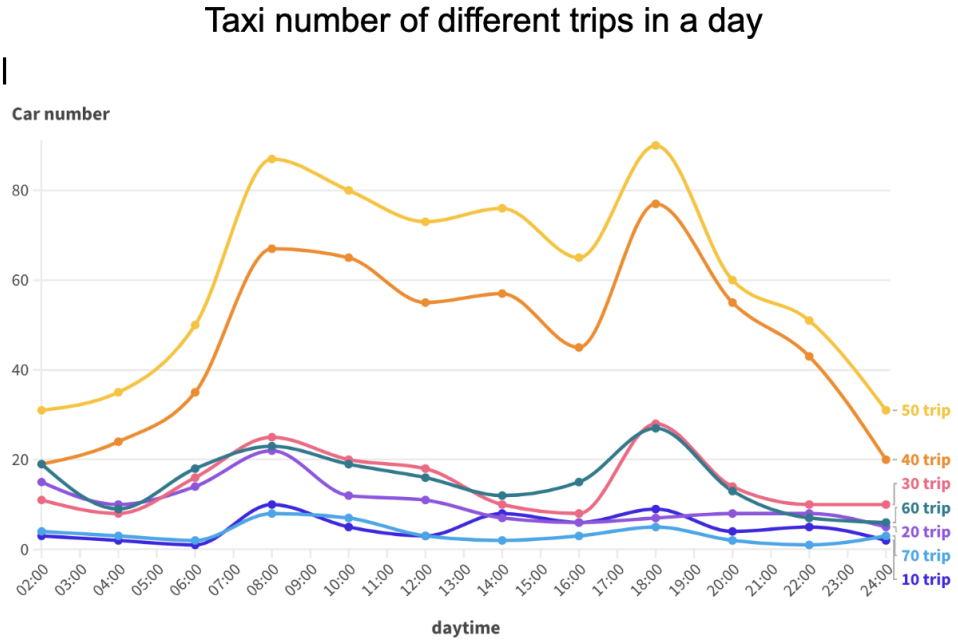


3D Surface Plot of Cluster Centers and Average Distances

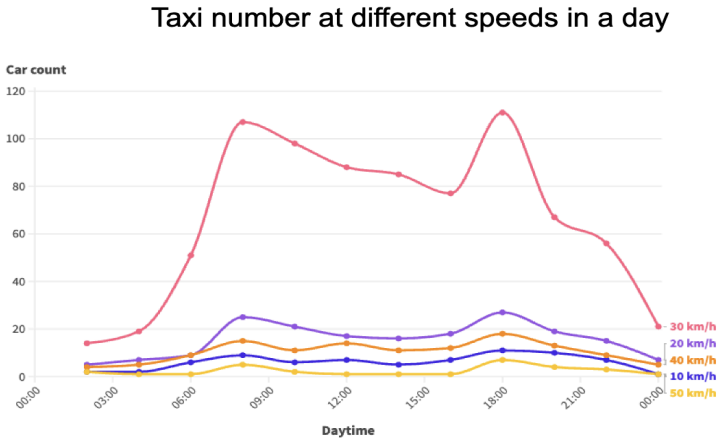


HTML for 3D chart: https://tonyc793.github.io/dv_show

5. Broken Line Analysis

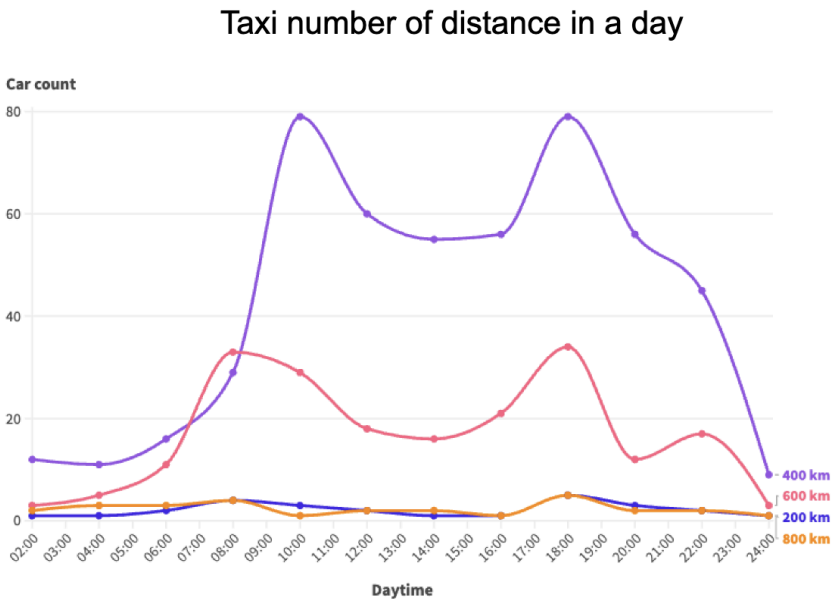


As shown in the graph, taxis with passenger counts of 40 and 50 are the most frequent, nearly three times as much as other passenger counts. Taxis with passenger counts of 70 and 10 are the least frequent. Throughout the day, we observe peak taxi usage at 8 AM and 6 PM, likely due to rush hours for work. Taxi passenger counts start to decline noticeably at 10 PM and continue until 4 AM, which is everyone's resting period.



From the graph, it can be observed that taxi drivers' driving speeds are mostly around 30 km/h, with few taxis either below or above this speed. The numbers decrease noticeably at speeds lower or higher than this threshold. Peak taxi

usage occurs from 8 AM to 9 AM and in the evening at 6 PM, making these times when the highest number of taxis are on the road. After 10 PM, the number of taxis at all speeds starts to decline as people finish work and head home to rest. Taxi numbers begin to increase again starting from 4 AM the next day.



As shown in the graph, taxis traveling a distance of 400 km are the most common throughout the day, followed by those traveling 600 km. Peak taxi usage begins at 8 AM, with a slight decrease in numbers around noon, which serves as a buffer time. Taxi numbers start to rise again from 5 PM and peak at 6 PM. After 6 PM, they enter a plateau period until 10 PM. Taxi numbers sharply decline after 10 PM and continue to do so until 5 AM the following day.

VIII. Evaluation

Evaluation: What did you learn about the data by using your visualizations? How did you answer your questions? How well does your visualization work, and how could you further improve it?

1. What did you learn about the data by using your visualizations?

The clusters of taxi activity revealed through our analysis highlighted strategic locations for taxi availability, which could help taxi companies optimize their fleet management. The time-lapse visualizations of taxi movements provided clear

illustrations of traffic flow at different times, identifying potential bottlenecks and optimal routes.

2. How did you answer your questions?

We determined peak traffic times and locations, suggesting optimal times for taxi availability to meet high demand. External factors like weather were observed to have a notable impact on travel patterns, with slower traffic flow and changes in pick-up points during adverse weather conditions.

3. How well does your visualization work, and how could you further improve it?

Our visualizations successfully displayed complex geographical and temporal data in an accessible format, allowing users to interact with the data and explore different aspects of taxi traffic dynamically.

Future enhancements could include predictive analytics features, using machine learning models to forecast taxi demand based on historical data and external factors such as weather and city events.

IX. Reference

1. Hochmair, Hartwig. (2016). Spatiotemporal Pattern Analysis of Taxi Trips in New York City. Transportation Research Record: Journal of the Transportation Research Board, 2542, 45-56. DOI: 10.3141/2542-06.

2. Kumar, Pratyush & Singh, Varun. (2023).Spatial-Temporal Analysis of Urban Mobility using Taxi Dataset. DOI: 10.21203/rs.3.rs-3630229/v1.

3. https://chriswhong.com/open-data/taxitechblog-2-leaflet-d3-and-other-frontend-fun/

4. https://cambridge-intelligence.com/visualizingnyc-taxi-cab-data/

5. https://sizmw.github.io/nyc-taxi-vis/

6. https://www.heavy.ai/demos/taxis

7. https://chriswhong.github.io/nyctaxi/

8. https://transdim.github.io/dataset/NYC-taxi/

9. https://medium.com/@muhammadaris10/nyctaxi-trip-data-analysis-45ecfdcb6f91

10.https://c2smart.engineering.nyu.edu/wpcontent/uploads/69A3551747124-Impact-of Ride-sharing-in-New-York-City.pdf

11. https://en.wikipedia.org/wiki/OpenStreetMap

12. https://wiki.openstreetmap.org/wiki/UsingOpenStreetMap

13. Mobility_Analysis_shenzhen_taxi (illinois.edu)