Name: Akanksha Shinde   Class: D15C   Roll No: 53

Experiment No 2

Perform following data visualization and exploration on your selected dataset.

Theory -

Data visualization and exploration are essential steps in understanding dataset characteristics, identifying patterns, and detecting anomalies. It involves statistical summaries, graphical representations, and correlation analysis. Exploratory Data Analysis (EDA) helps in making informed decisions before applying machine learning or hypothesis testing. Common visualizations include histograms, scatter plots, box plots, and heatmaps. Feature relationships can be analyzed using correlation matrices. Proper data exploration ensures better insights, improved model accuracy, and meaningful conclusions.

Matplotlib is a low-level library for creating static, animated, and interactive visualizations

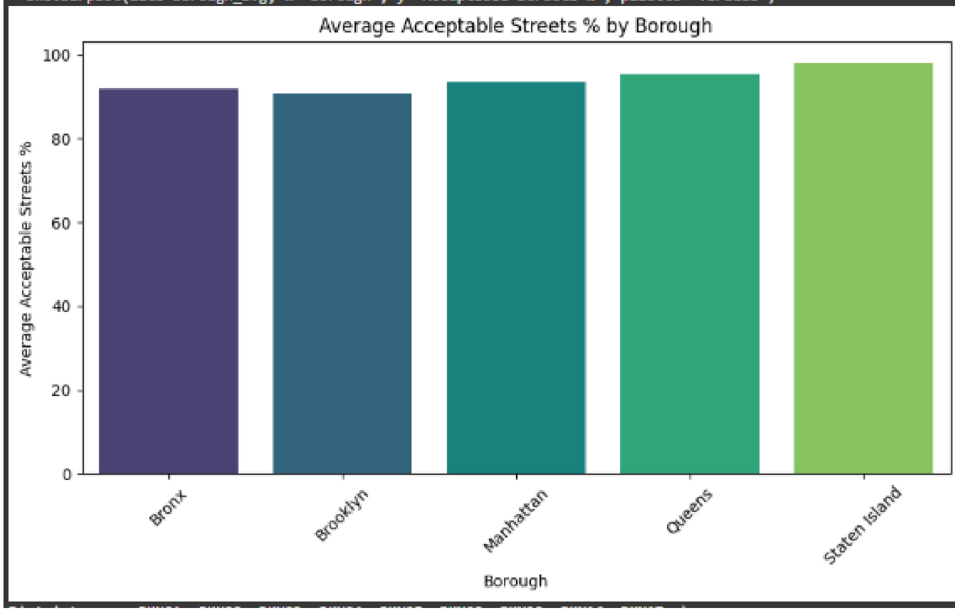
Seaborn is built on top of Matplotlib, providing a high-level interface for statistical graphics with better aesthetics.

1. Create bar graph, contingency table using any 2 features.

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

# Load dataset
file_path = "/content/scorecard.csv"
df = pd.read_csv(file_path)

# Drop missing values
df_clean = df.dropna(subset=["Borough", "Acceptable Streets %"])

# Calculate mean values separately
borough_avg = df_clean.groupby("Borough")["Acceptable Streets %"].mean().reset_index()

# Bar plot: Average "Acceptable Streets %" by Borough
plt.figure(figsize=(10, 5))
sns.barplot(data=borough_avg, x="Borough", y="Acceptable Streets %", palette="viridis")
plt.xticks(rotation=45)
plt.title("Average Acceptable Streets % by Borough")
plt.xlabel("Borough")
plt.ylabel("Average Acceptable Streets %")
plt.show()

# Contingency table: Borough vs. District
contingency_table = pd.crosstab(df["Borough"], df["District"])
print(contingency_table)
```

Name: Akanksha Shinde Class: D15C Roll No: 53

```
sns.barplot(data=borough_avg, x="Borough", y="Acceptable Streets %", palette="viridis")
```

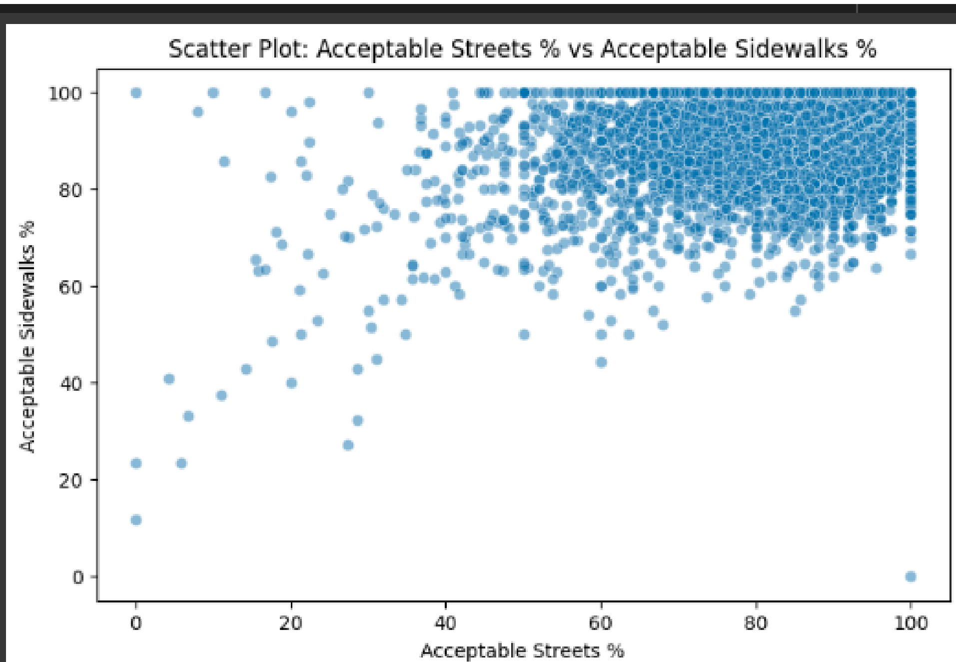



```
District       BKN01  BKN02  BKN03  BKN04  BKN05  BKN08  BKN09  BKN16  BKN17  \
Borough
Bronx              0      0      0      0      0      0      0      0      0
Brooklyn        1020    816   1020    612    816    612    612    770   1020
Manhattan          0      0      0      0      0      0      0      0      0
Queens             0      0      0      0      0      0      0      0      0
Staten Island      0      0      0      0      0      0      0      0      0

District       BKS06  ...  QW01  QW02  QW03  QW04  QW05  QW06  QW09  SI01  \
Borough               ...
Bronx              0  ...     0     0     0     0     0     0     0     0
Brooklyn        1021  ...     0     0     0     0     0     0     0     0
Manhattan          0  ...     0     0     0     0     0     0     0     0
Queens             0  ...  1224   612   612   612  1020   408   816     0
Staten Island      0  ...     0     0     0     0     0     0     0   816

District       SI02  SI03
Borough
Bronx             0     0
Brooklyn          0     0
Manhattan         0     0
Queens            0     0
Staten Island   816  1598

[5 rows x 59 columns]
```

Name: Akanksha Shinde   Class: D15C   Roll No: 53

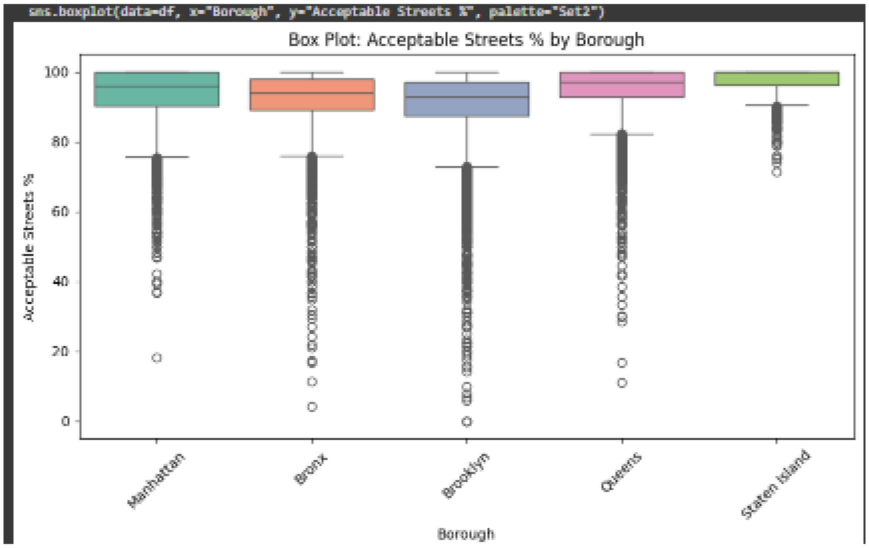
2. Plot Scatter plot, box plot, Heatmap using seaborn.

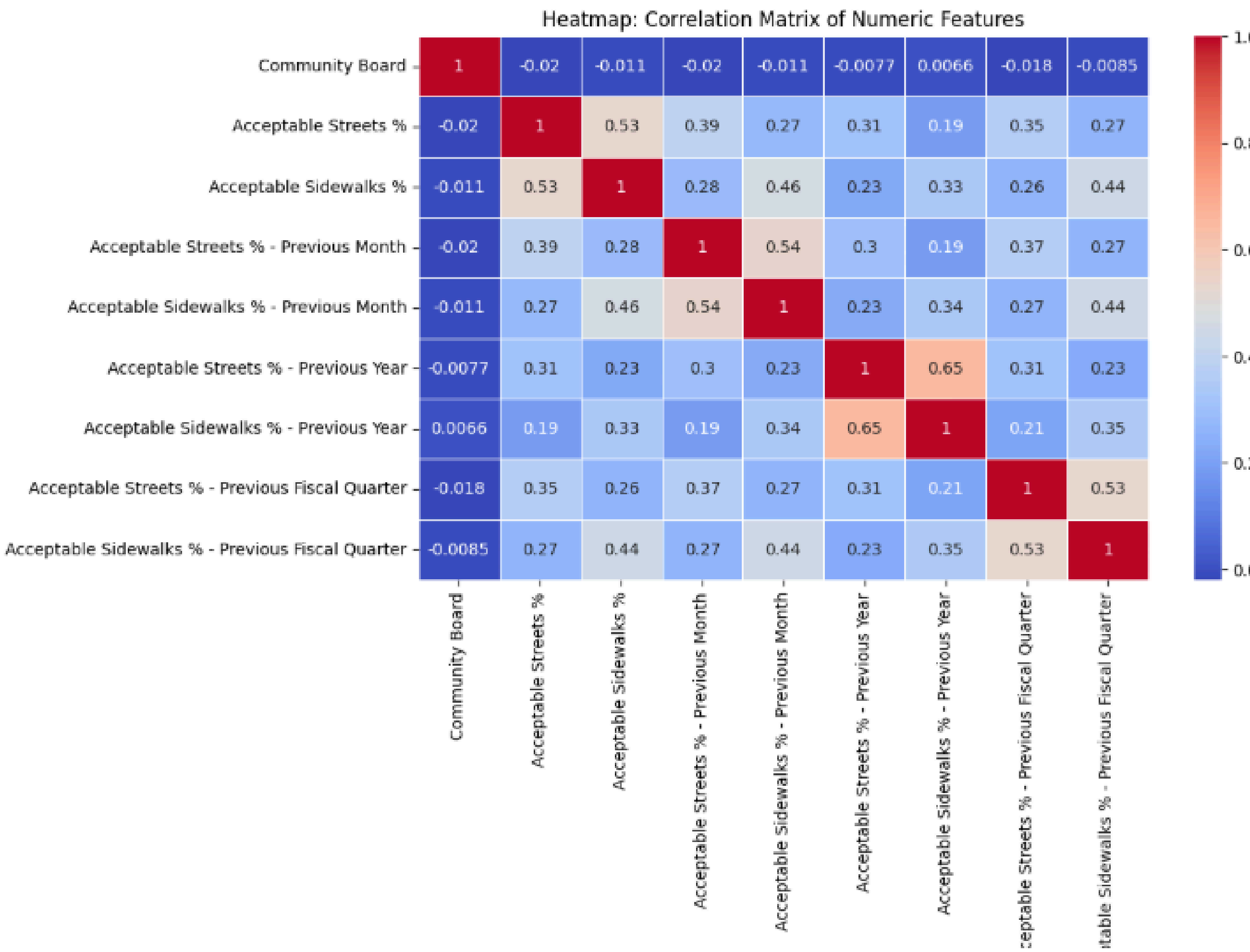
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load dataset
file_path = "/content/scorecard.csv"  # Update with correct path
df = pd.read_csv(file_path)

# Drop missing values for relevant columns
df_numeric = df.dropna(subset=["Acceptable Streets %", "Acceptable Sidewalks %"])

# Scatter Plot: Acceptable Streets % vs Acceptable Sidewalks %
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df_numeric, x Loading... le Streets %", y="Acceptable Sidewalks %", alpha=0.5)
plt.title("Scatter Plot: Acceptable Streets % vs Acceptable Sidewalks %")
plt.xlabel("Acceptable Streets %")
plt.ylabel("Acceptable Sidewalks %")
plt.show()

# Box Plot: Distribution of Acceptable Streets % by Borough
plt.figure(figsize=(10, 5))
sns.boxplot(data=df, x="Borough", y="Acceptable Streets %", palette="Set2")
plt.xticks(rotation=45)
plt.title("Box Plot: Acceptable Streets % by Borough")
plt.xlabel("Borough")
plt.ylabel("Acceptable Streets %")
plt.show()

# Heatmap: Correlation matrix of numeric columns
plt.figure(figsize=(10, 6))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap="coolwarm", linewidths=0.5)
plt.title("Heatmap: Correlation Matrix of Numeric Features")
plt.show()
```




```
<ipython-input-4-002e71a49308>:22: FutureWarning:
```

Name: Akanksha Shinde   Class: D15C   Roll No: 53

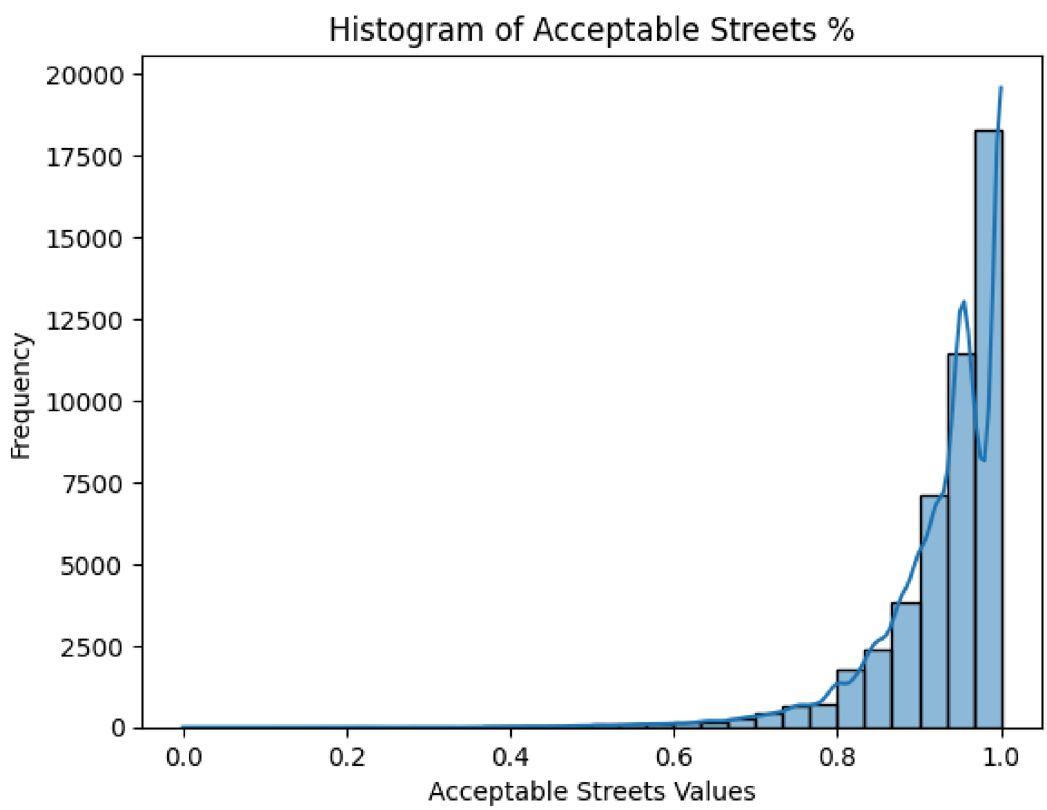




Name: Akanksha Shinde Class: D15C Roll No: 53

3. Create histogram and normalized Histogram.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Example: Histogram of 'Acceptable Streets %'
sns.histplot(df['Acceptable Streets %'], bins=30, kde=True)
plt.title("Histogram of Acceptable Streets %")
plt.xlabel("Acceptable Streets Values")
plt.ylabel("Frequency")
plt.show()
```



Name: Akanksha Shinde   Class: D15C   Roll No: 53

4. Describe what this graph and table indicates.

Table Description:

1. The table represents data for districts across five New York City boroughs: Bronx, Brooklyn, Manhattan, Queens, and Staten Island.
2. Each borough is represented in rows, and each district within the boroughs is represented in columns (e.g., **BKN01**, **QW01**, **SI01**).
3. The **values** in the cells (e.g., **0**, **612**, **1020**) correspond to a specific metric (e.g., counts, measurements, or scores).
4. **Zero values** in several boroughs (Bronx, Manhattan, Queens, Staten Island) may indicate the absence of data or lack of measurement in those districts.
5. **Brooklyn** shows significant values (e.g., **1020**, **816**) in several districts, potentially indicating higher activity, population, or other metrics.

Graph Description:

1. The graph likely visualizes the distribution of these values across boroughs and districts.
2. **Bar graphs** or **heat maps** could be used to display the varying values across the districts of each borough.
3. In a **heat map**, Brooklyn's districts with higher values (e.g., **BKN01**, **BKN02**) would be represented with **high-intensity colors**, while other boroughs would have **lower-intensity or neutral colors** (e.g., Bronx, Manhattan, Staten Island).
4. If the graph is a **bar graph**, it might show multiple bars for each borough, with **Brooklyn** having the tallest bars in its districts, while other boroughs (like the Bronx or Queens) may have shorter or no bars.
5. The graph would visually highlight the stark differences between Brooklyn and the other boroughs, reinforcing the dominance of Brooklyn in this dataset.

Name: Akanksha Shinde   Class: D15C   Roll No: 53

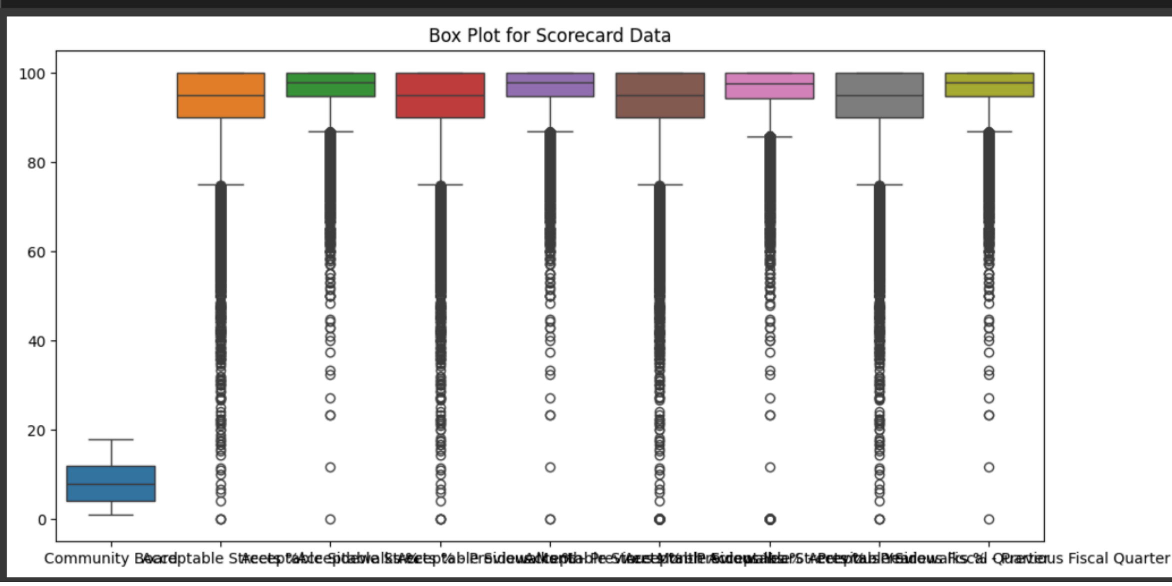
5. Handle outlier using box plot and Interquartile range.

File Edit View Insert Runtime Tools Help

+ Code + Text

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

# Load the dataset
df = pd.read_csv("scorecard.csv")

# Create a box plot to visualize outliers
plt.figure(figsize=(12, 6))
sns.boxplot(data=df)
plt.title('Box Plot for Scorecard Data')
plt.show()

# Calculate Q1 (25th percentile) and Q3 (75th percentile)
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)

# Calculate the IQR (Interquartile Range)
IQR = Q3 - Q1

# Determine the lower and upper bounds for outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Remove outliers from the dataset
df_no_outliers = df[(df >= lower_bound) & (df <= upper_bound)].dropna()

# Create a box plot after removing outliers
plt.figure(figsize=(12, 6))
sns.boxplot(data=df_no_outliers)
plt.title('Box Plot After Removing Outliers')
plt.show()

# Display the cleaned dataset (if needed)
# print(df_no_outliers)
```

+ Code + Text



Name: Akanksha Shinde Class: D15C Roll No: 53

Conclusion -

The data visualization and exploration provided key insights into feature relationships and distributions. Bar graphs and contingency tables revealed patterns, while scatter plots, box plots, and heatmaps highlighted correlations and outliers. Histograms helped analyze data distribution, detecting skewness or uniformity. Interpretation of these visualizations allowed us to spot trends and anomalies. Outlier handling using box plots and IQR ensured cleaner data for better analysis.