Visulization library

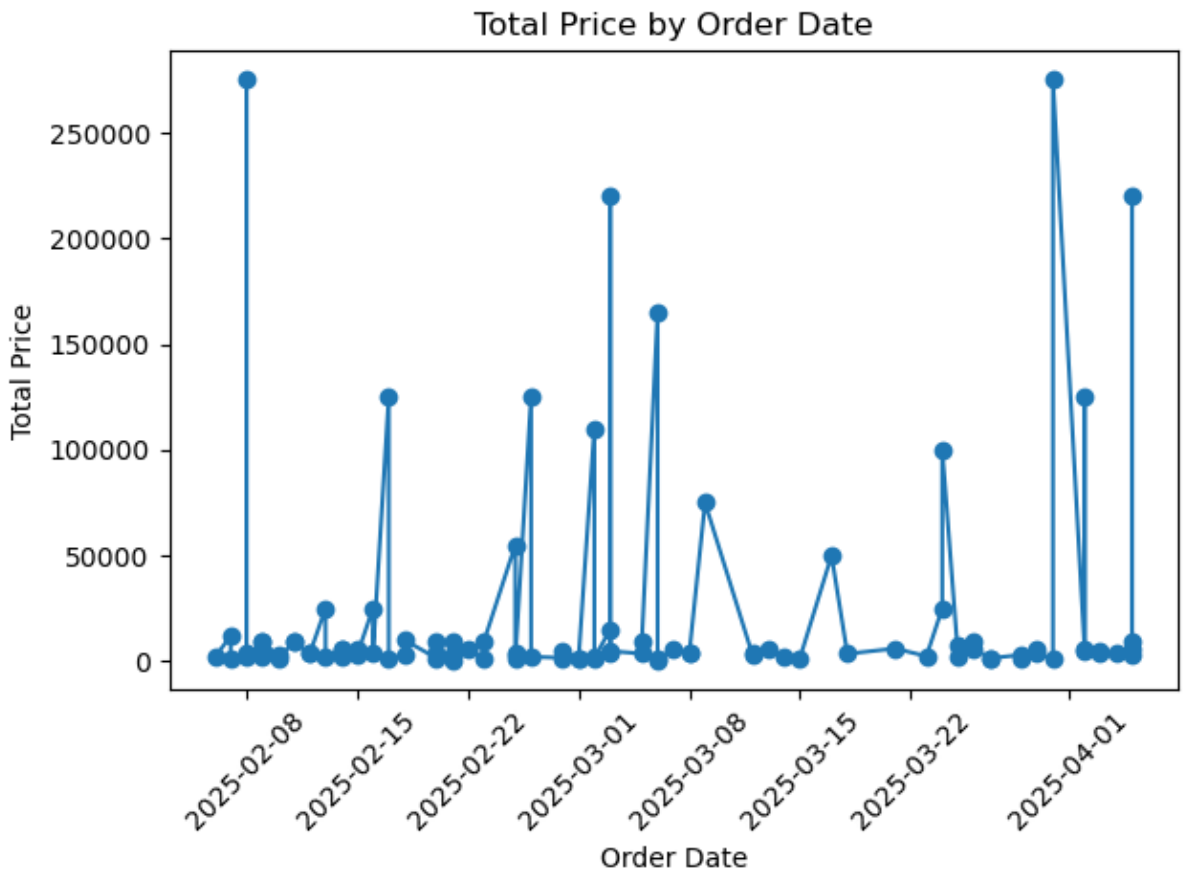
1. **Matplotlib**

   - It is a popular Python library used for creating static, animated, and interactive visualizations. It provides a wide variety of plots, such as line plots, bar plots, scatter plots, histograms, and more. Matplotlib is highly customizable and gives users fine control over plot elements (e.g., colors, labels, axes). It is commonly used in data analysis to visualize data patterns and trends.

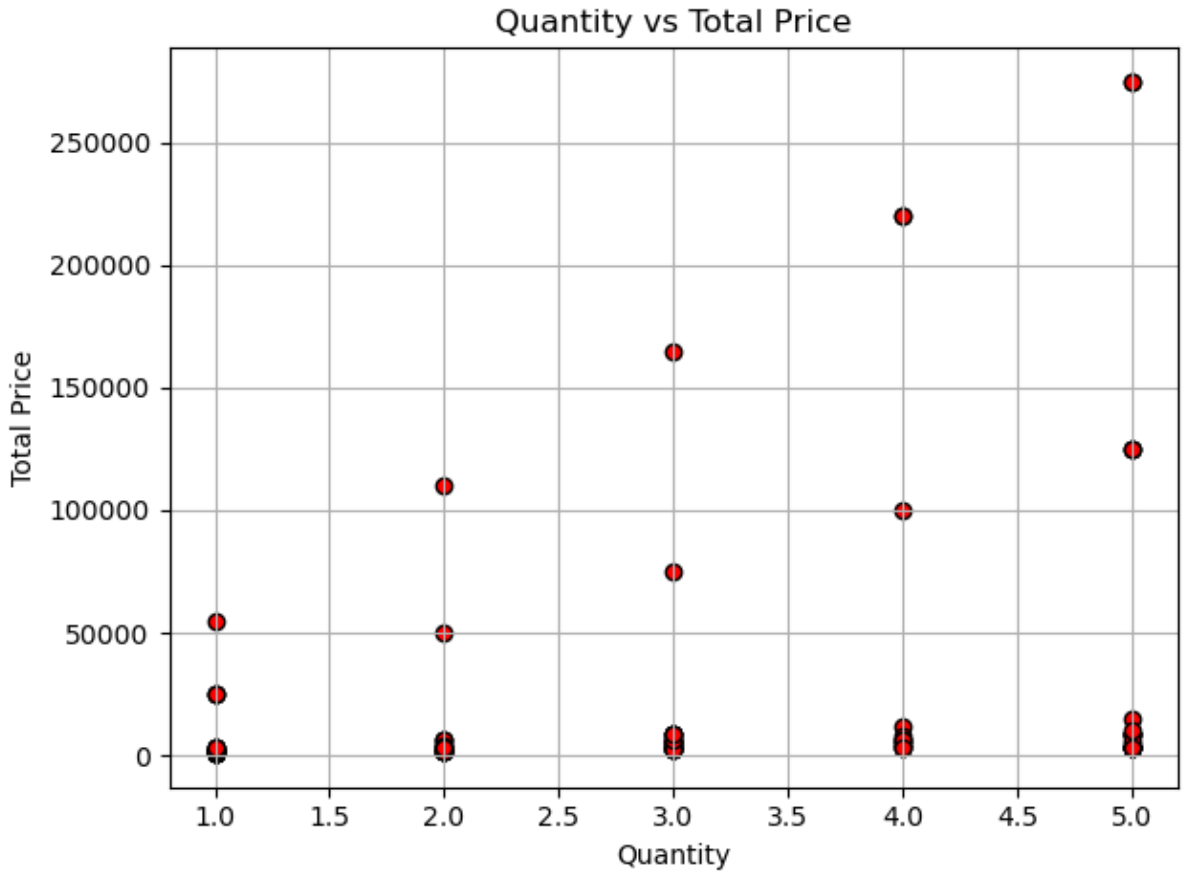
2. **Seaborn**

   - It is built on top of Matplotlib and provides a high-level interface for creating attractive and informative statistical graphics. It simplifies complex visualizations by offering functions to create heatmaps, violin plots, box plots, and more, with better default aesthetics than Matplotlib. Seaborn also integrates well with Pandas DataFrames and is often preferred for visualizing statistical relationships between data points.

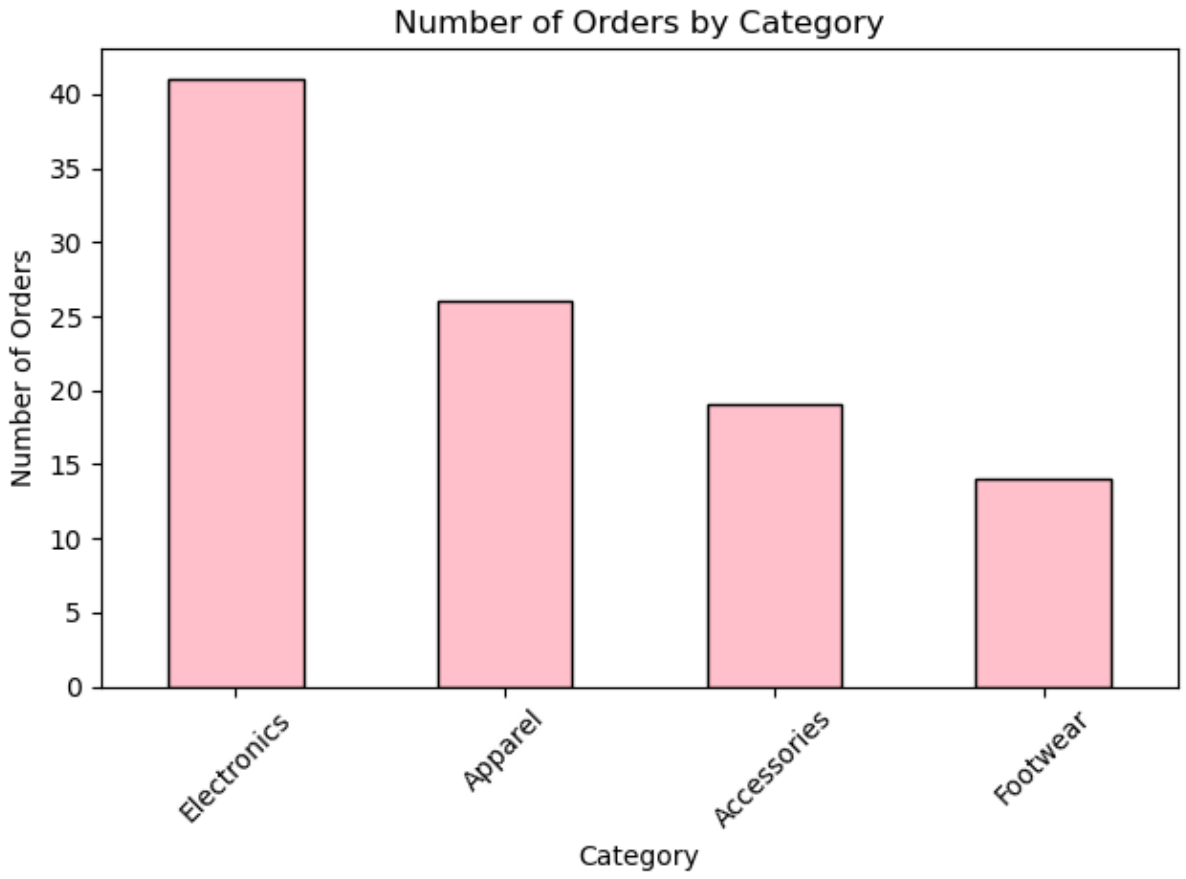
```python
import pandas as pd
import matplotlib.pyplot as plt

# Load the Excel file
df_100 = pd.read_excel("F:\ShadowFox\Level = 1\Online_Shopping.xlsx")--


# 1. Line Plot: Total Price by Order Date(Sorted)
df_line = df_100.sort_values("Order Date")
plt.figure()
plt.plot(df_line["Order Date"], df_line["Total Price"], marker='o', linestyle='-')
plt.title("Total Price by Order Date")
plt.xlabel("Order Date")
plt.ylabel("Total Price")
plt.xticks(rotation=45)
plt.tight_layout()

plt.show()
```



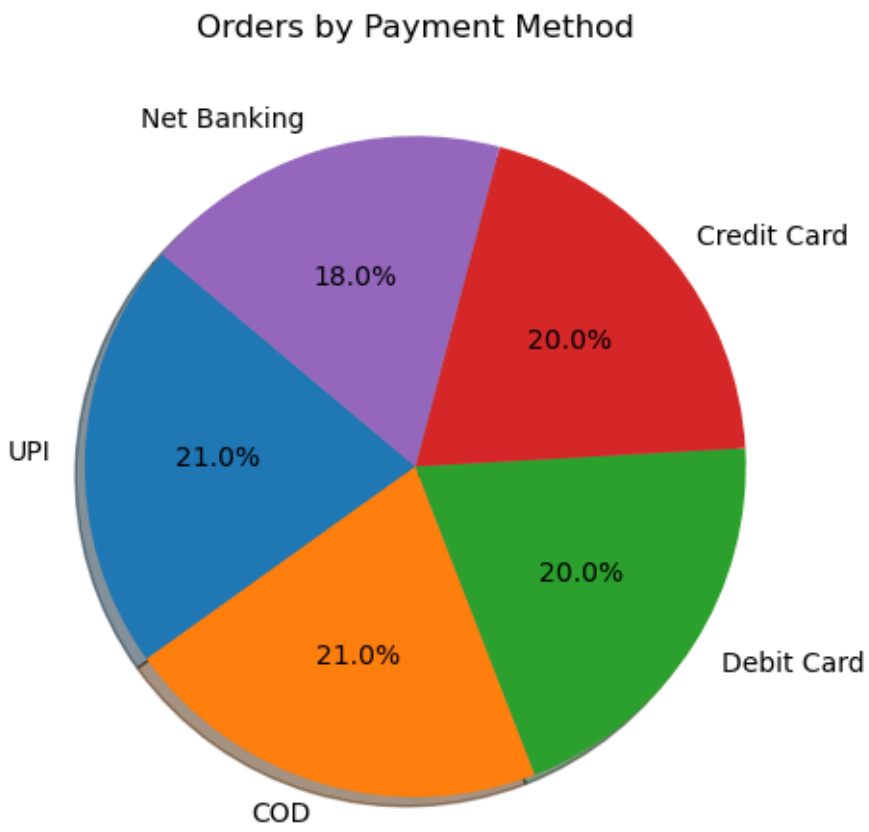
```python
# 2. Scatter Plot: Quantity vs Total Price
plt.figure()
plt.scatter(df_100["Quantity"], df_100["Total Price"], c='red', edgecolors='k')
plt.title("Quantity vs Total Price")
plt.xlabel("Quantity")
plt.ylabel("Total Price")
plt.grid(True)
plt.tight_layout()

plt.show()
```



```python
# 3. Bar Chart: Count of Orders by Category
category_counts = df_100["Category"].value_counts()

plt.figure()
category_counts.plot(kind='bar', color='pink', edgecolor='black')
plt.title("Number of Orders by Category")
plt.xlabel("Category")
plt.ylabel("Number of Orders")
plt.xticks(rotation=45)
plt.tight_layout()
```



```python
# 4. Pie Chart: Orders by Payment Method
payment_counts = df_100["Payment Method"].value_counts()

plt.figure()
payment_counts.plot(kind='pie', autopct='%1.1f%%', startangle=140, shadow=True)
plt.title("Orders by Payment Method")
plt.ylabel("")  # Hide y-label for a cleaner pie
plt.tight_layout()

plt.show()
```



```
# 5. Histogram: Distribution of Total Prices
plt.figure()
plt.hist(df_100["Total Price"], bins=10, color='skyblue', edgecolor='black')
plt.title("Histogram of Total Prices")
plt.xlabel("Total Price")
plt.ylabel("Frequency")
plt.tight_layout()

plt.show()
```