Superstore sales
Data Analysis Using: Python

Group Code: ISM1_DAT1_M1e

Track : Data Analysis Specialist

Technical Instructor: Soha Nagy

Team Members:

Nancy Ahmed Mohamed

Fatma Mohamed Elsayed

Nehal Nabil Ibrahim

Nourhan Mohamed Mohamed

Nardine Emad Salama

Table of Content

About The Company

Superstore is a fictional retail company based in the United States. They specialize in selling furniture, office supplies, and technology products. The objective is to identify weaknesses and opportunities within their business, and help them enhance their business growth and profitability.

Data Description

Superstore product dataset containing sales and profit that group by into furniture, office supplies and technology. The dataset used for this analysis is sourced from the Superstore dataset, which contains information about sales transactions, customer demographics, and product details. This dataset encompasses a wide range of information, including order specifics, geographical data, and product-related data. There are no missing values or any irrelevant data types and values.
This project involves:

- Exploring and cleaning the dataset to ensure data quality.
- Performing exploratory data analysis (EDA) to uncover trends and insights.
- Visualizing sales trends, including seasonal patterns and product performance.

Questions

- What is the most important segment?
- What is the trend of sales over time?
- Why did consumers have the highest sales?
- What are the highest and least selling categories and subcategories of products?
- Highest regions and cities in sales?
- What is the most preferred shipping mood?
- Why is Standard mode of shipping the highest?
- Did ship duration affect order frequency?
- Top 10 customers
- Why did sales decrease during 2016 and increase During 2018?

Data Cleaning

First, to answer that question we should do Data Cleaning that involves removing duplicate data and handling null (missing values). The first step was to import the libraries and read the dataset and read CSV file as shown in the figure

```python
# import required libraries
import numpy as np
import pandas as pd # data processing, CSV file
import statistics # for calculations
import matplotlib.pyplot as plt #for pie chart
import seaborn as sns #for heatmap
import plotly.express as px #for bar chart
# Read the Excel file.
df = pd.read_csv("//content/Superstore Sales Dataset.csv")
```

- Only postal code has 11 missing values
- Replace null values with postal code of Burlington
- Change order and ship date data type into date type
- No duplicates

Data Exploration & Visualization

Data contains 9800 sales transactions that occurred from 2015 to 2018.

- (9800 row, 18 column)
- Data types: float64(2), int64(1), object(15)

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID
0	1	CA-2017-152156	08/11/2017	11/11/2017	Second Class	CG-12520
1	2	CA-2017-152156	08/11/2017	11/11/2017	Second Class	CG-12520
2	3	CA-2017-138688	12/06/2017	16/06/2017	Second Class	DV-13045
3	4	US-2016-108966	11/10/2016	18/10/2016	Standard Class	SO-20335
4	5	US-2016-108966	11/10/2016	18/10/2016	Standard Class	SO-20335

	Customer Name	Segment	Country	City	State
0	Claire Gute	Consumer	United States	Henderson	Kentucky
1	Claire Gute	Consumer	United States	Henderson	Kentucky
2	Darrin Van Huff	Corporate	United States	Los Angeles	California
3	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida
4	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida

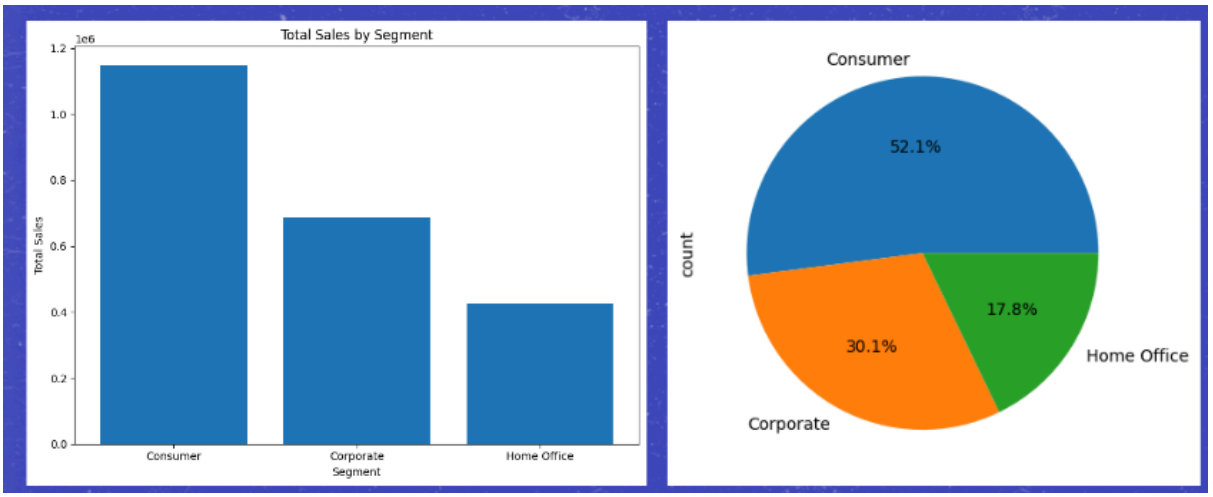
	Postal Code	Region	Product ID	Category	Sub-Category
0	42420.0	South	FUR-BO-10001798	Furniture	Bookcases
1	42420.0	South	FUR-CH-10000454	Furniture	Chairs
2	90036.0	West	OFF-LA-10000240	Office Supplies	Labels
3	33311.0	South	FUR-TA-10000577	Furniture	Tables
4	33311.0	South	OFF-ST-10000760	Office Supplies	Storage

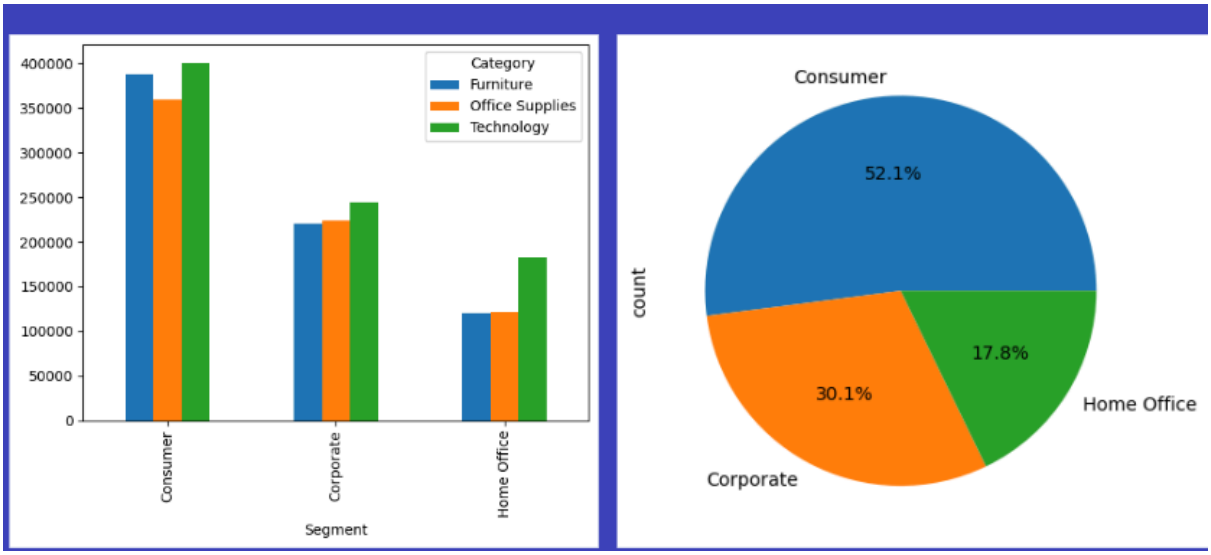
	Product Name	Sales
0	Bush Somerset Collection Bookcase	261.9600
1	Hon Deluxe Fabric Upholstered Stacking Chairs,...	731.9400
2	Self-Adhesive Address Labels for Typewriters b...	14.6200
3	Bretford CR4500 Series Slim Rectangular Table	957.5775
4	Eldon Fold 'N Roll Cart System	22.3680

```
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Row ID          9800 non-null   int64
 1   Order ID        9800 non-null   object
 2   Order Date      9800 non-null   object
 3   Ship Date       9800 non-null   object
 4   Ship Mode       9800 non-null   object
 5   Customer ID     9800 non-null   object
 6   Customer Name   9800 non-null   object
 7   Segment         9800 non-null   object
 8   Country         9800 non-null   object
 9   City            9800 non-null   object
 10  State           9800 non-null   object
 11  Postal Code     9789 non-null   float64
 12  Region          9800 non-null   object
 13  Product ID      9800 non-null   object
 14  Category        9800 non-null   object
 15  Sub-Category    9800 non-null   object
 16  Product Name    9800 non-null   object
 17  Sales           9800 non-null   float64
dtypes: float64(2), int64(1), object(15)
memory usage: 1.3+ MB
```

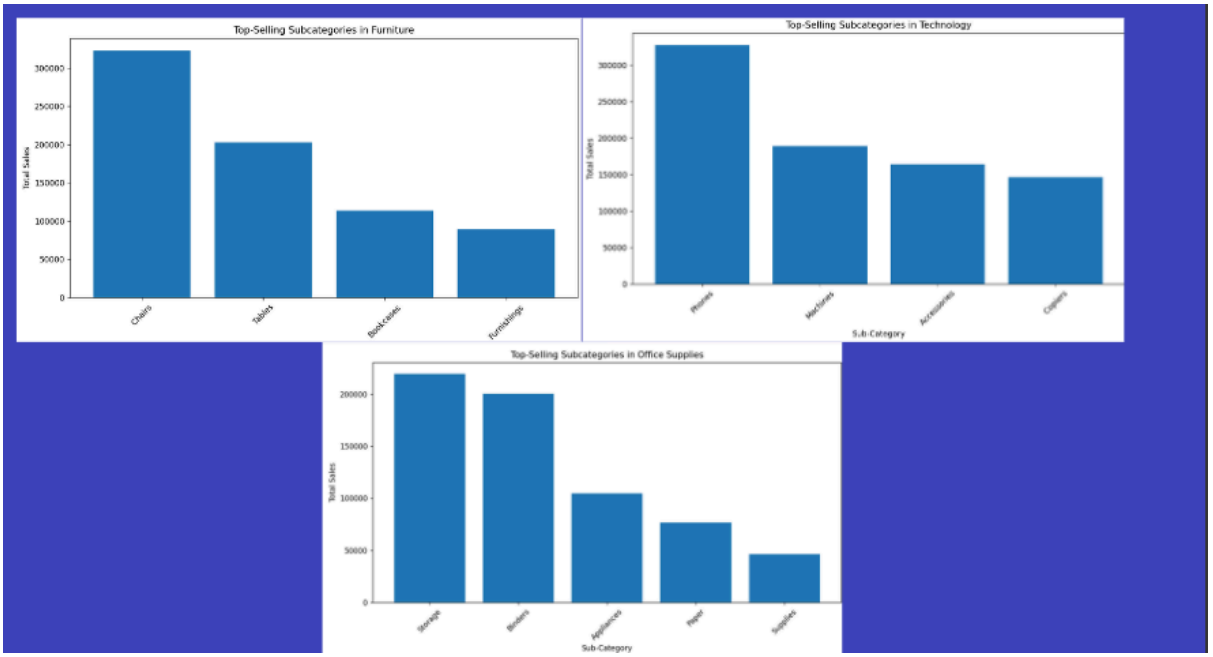
Order date and Ship date data type was changed to into date type

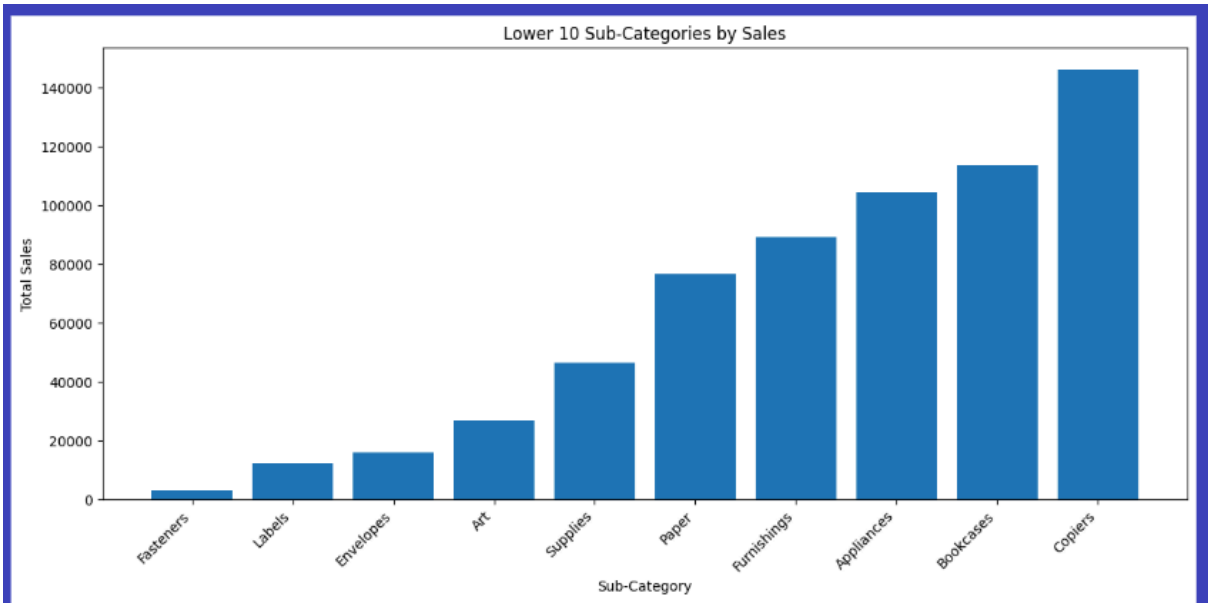
Results

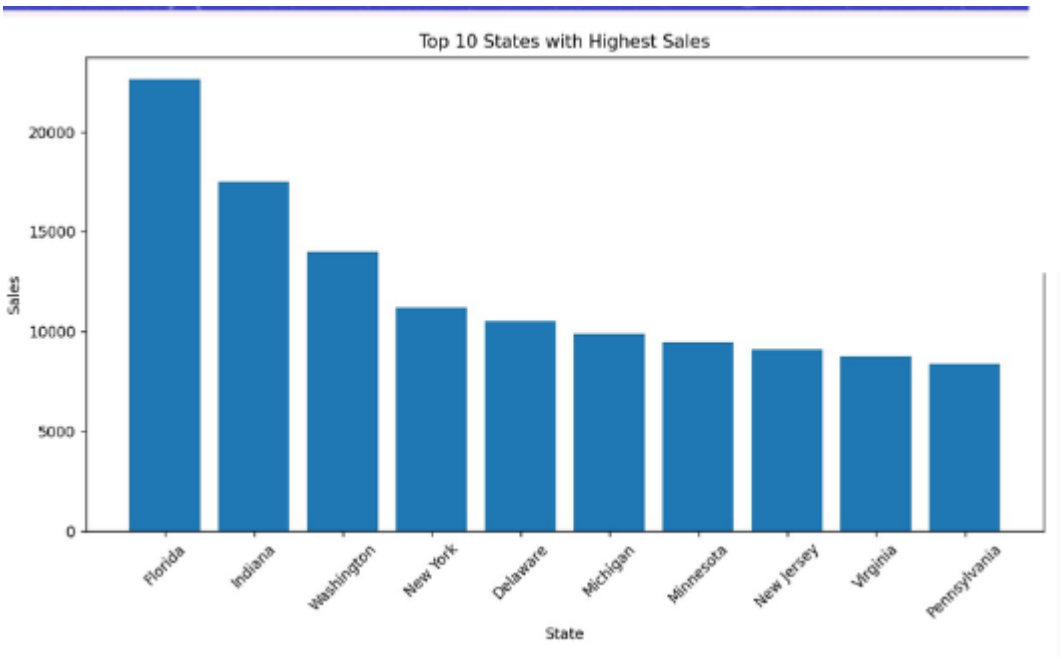


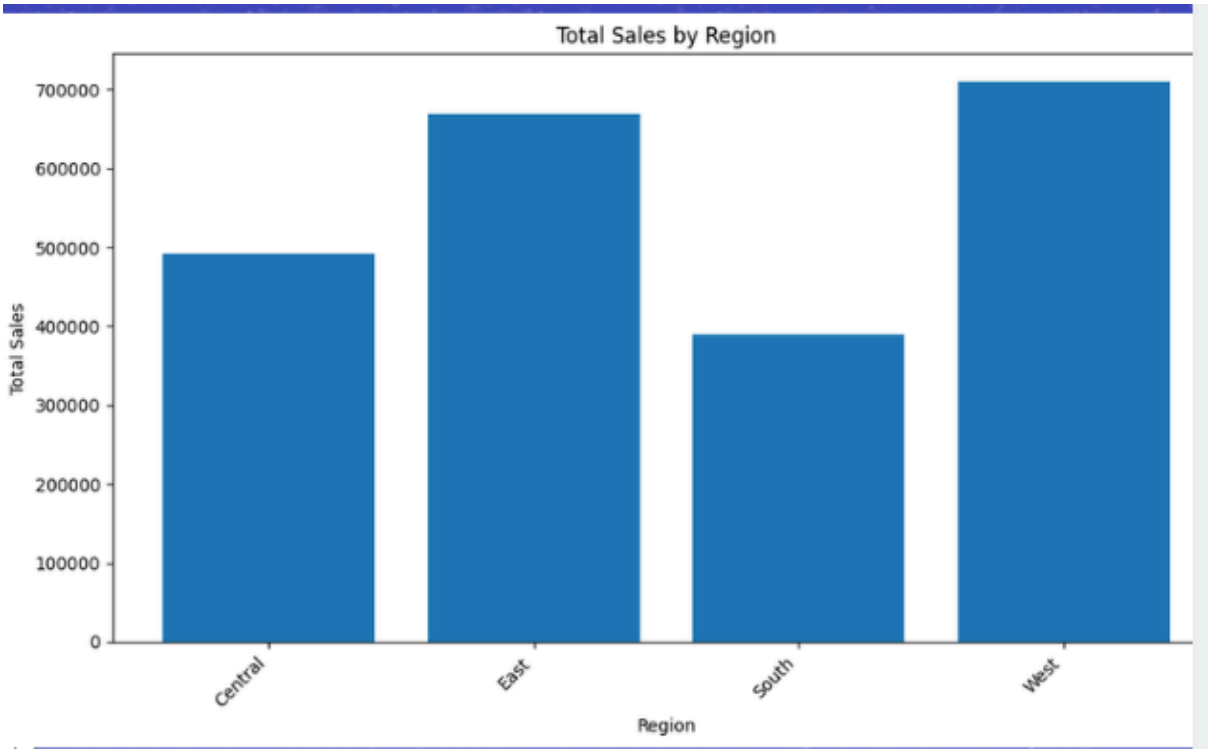

Category
Furniture
Office Supplies
Technology
400000
350000
300000
250000
200000
150000
100000
50000
0
Consumer
Corporate
Home Office
Segment
count
Consumer
52.1%
17.8%
30.1%
Home Office
Corporate

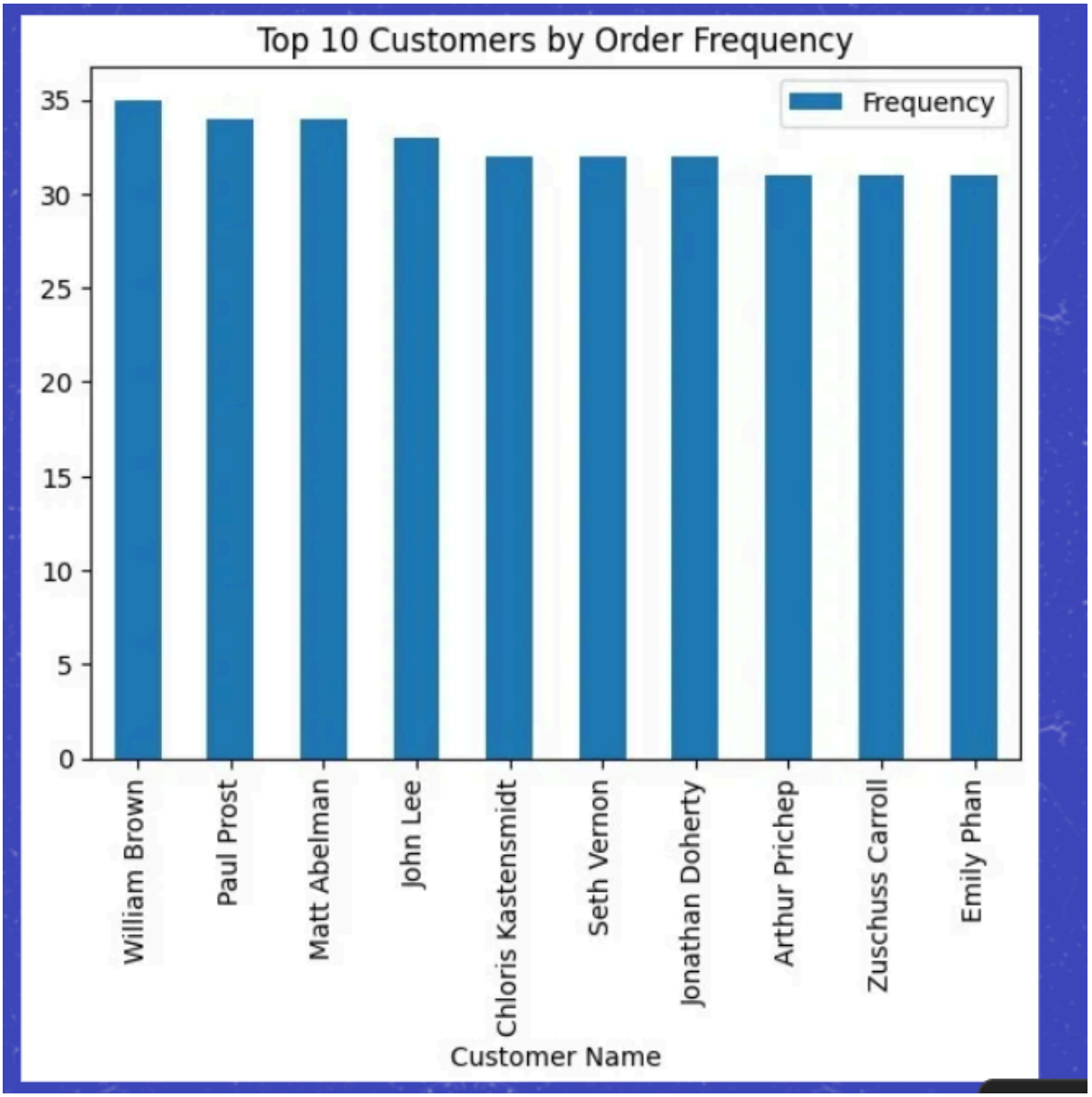

Product Category Sales Distribution
Furniture
32.2%
Office Supplies
31.2%
36.6%
Technology

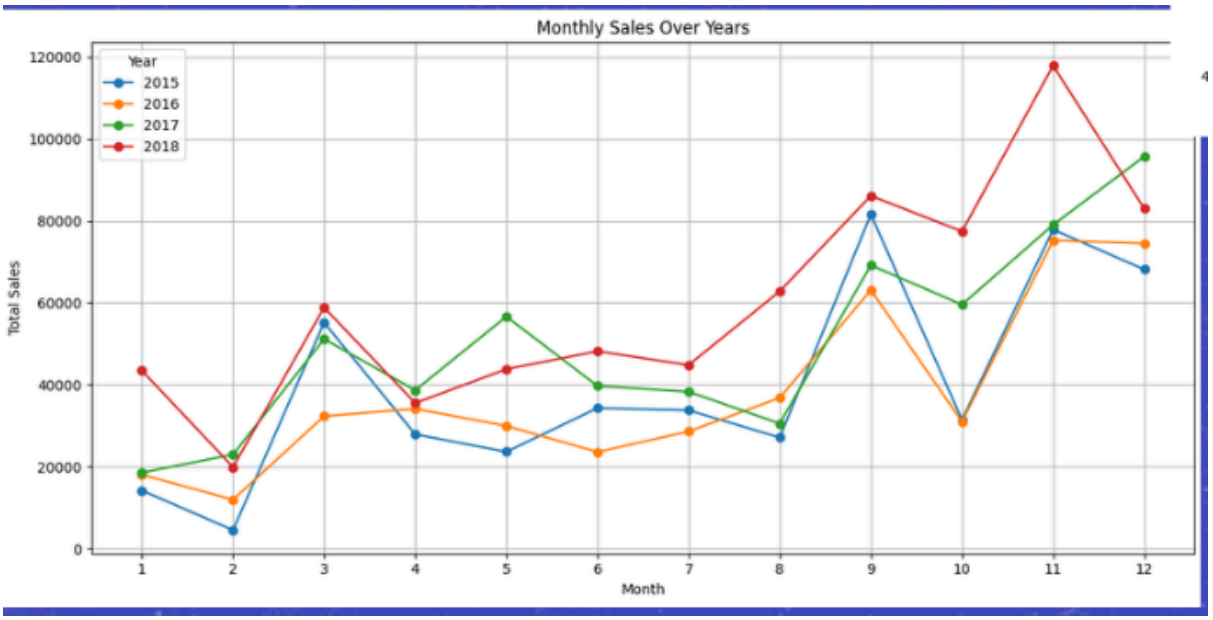

Top-Selling Subcategories in Furniture
Total Sales
Chairs
Tables
Bookcases
Furnishings
Top-Selling Subcategories in Technology
Total Sales
Phones
Machines
Accessories
Copiers
Sub-Category
Top-Selling Subcategories in Office Supplies
Total Sales
Storage
Binders
Appliances
Paper
Supplies
Sub-Category

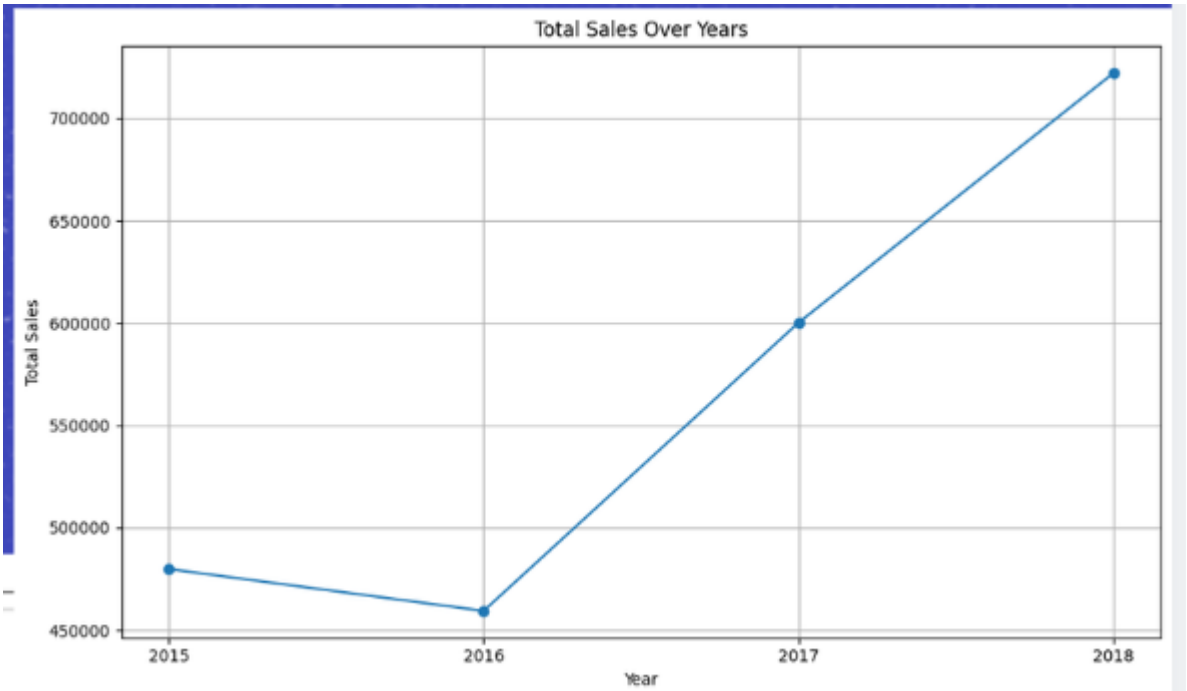

Lower 10 Sub-Categories by Sales
Total Sales
0
20000
40000
60000
80000
100000
120000
140000
Fasteners
Labels
Envelopes
Art
Supplies
Paper
Furnishings
Appliances
Bookcases
Copiers
Sub-Category

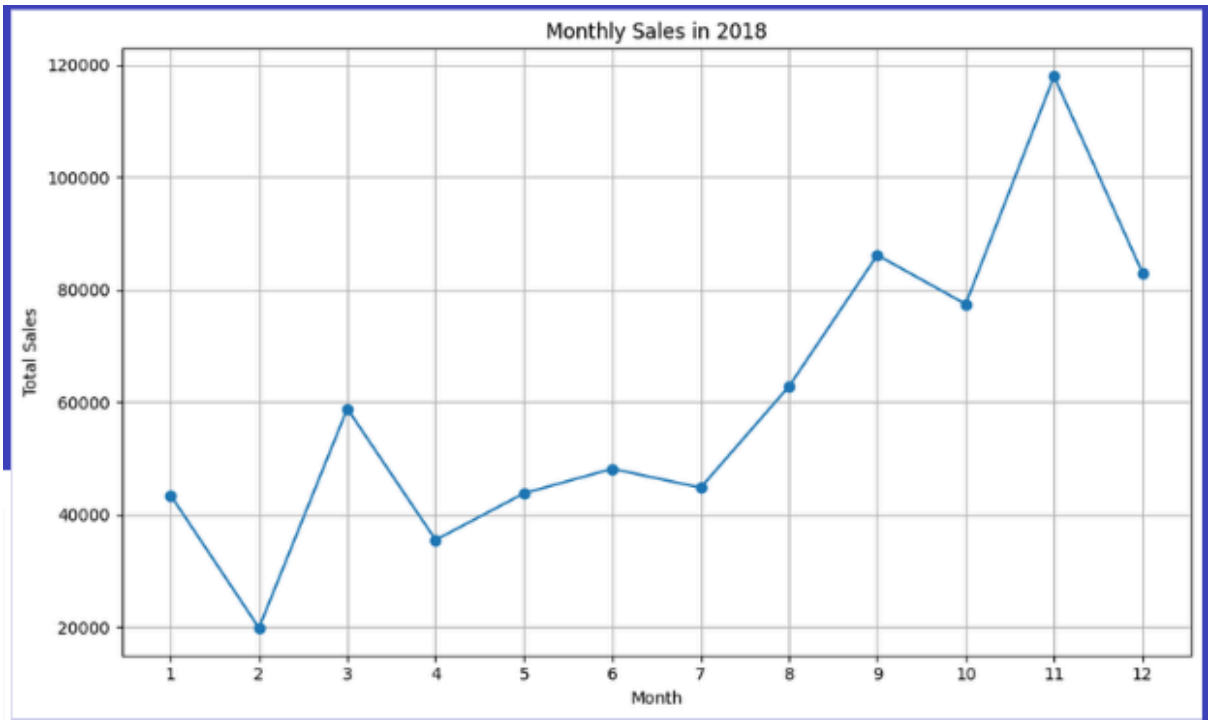

Top 10 States with Highest Sales
Sales
20000
15000
10000
5000
0
Florida
Indiana
Washington
New York
Delaware
Michigan
Minnesota
New Jersey
Virginia
Pennsylvania
State

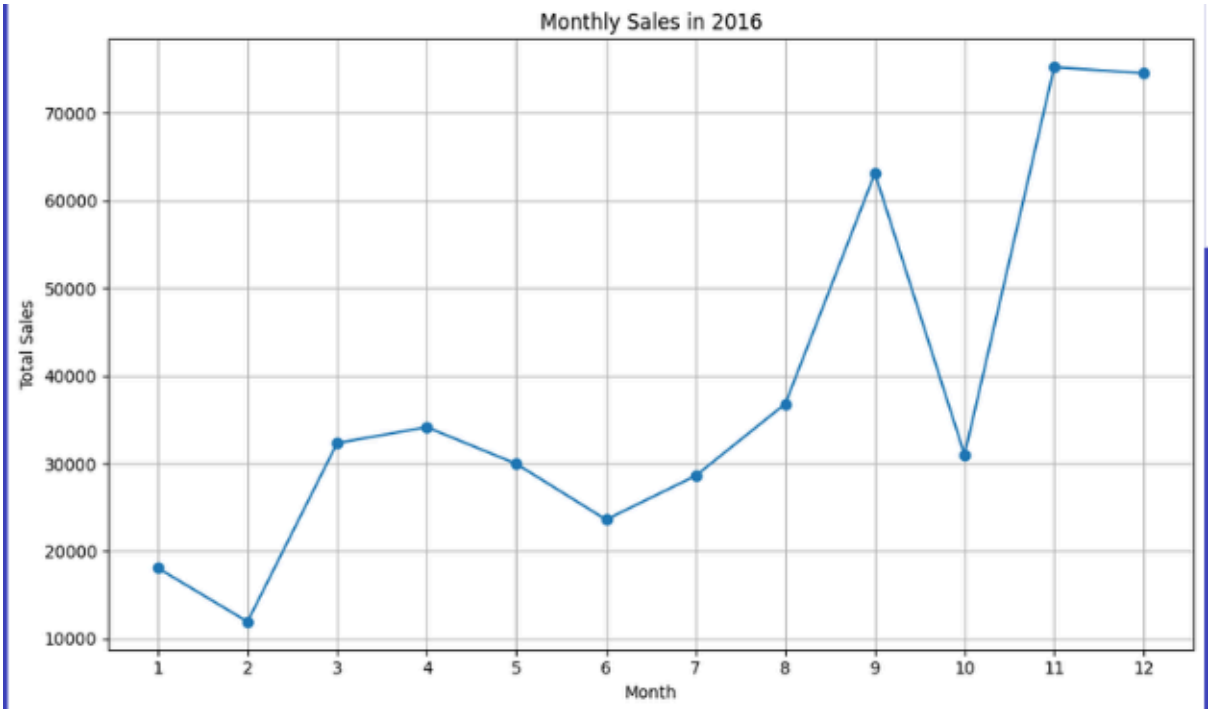

Total Sales by Region
Total Sales
700000
600000
500000
400000
300000
200000
100000
0
Central
East
South
West
Region

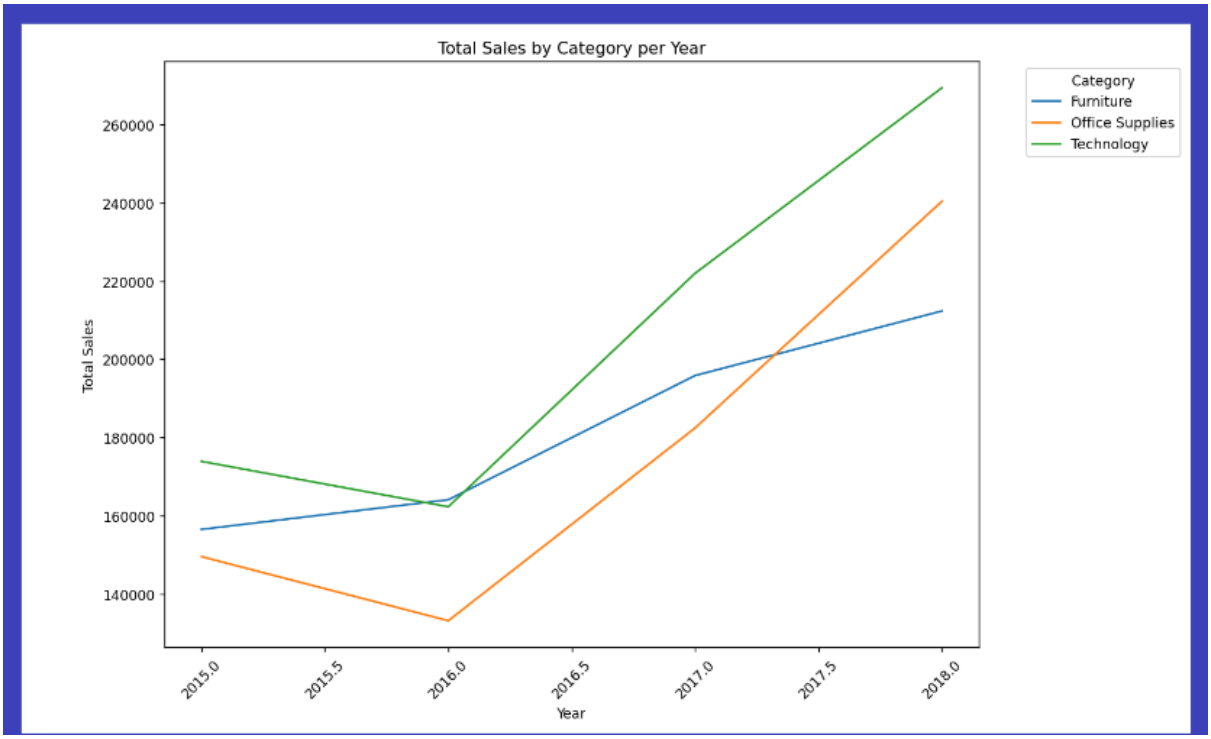

Top 10 Customers by Order Frequency
Frequency
35
30
25
20
15
10
5
0
William Brown
Paul Prost
Matt Abelman
John Lee
Chloris Kastensmidt
Seth Vernon
Jonathan Doherty
Arthur Prichep
Zuschuss Carroll
Emily Phan
Customer Name

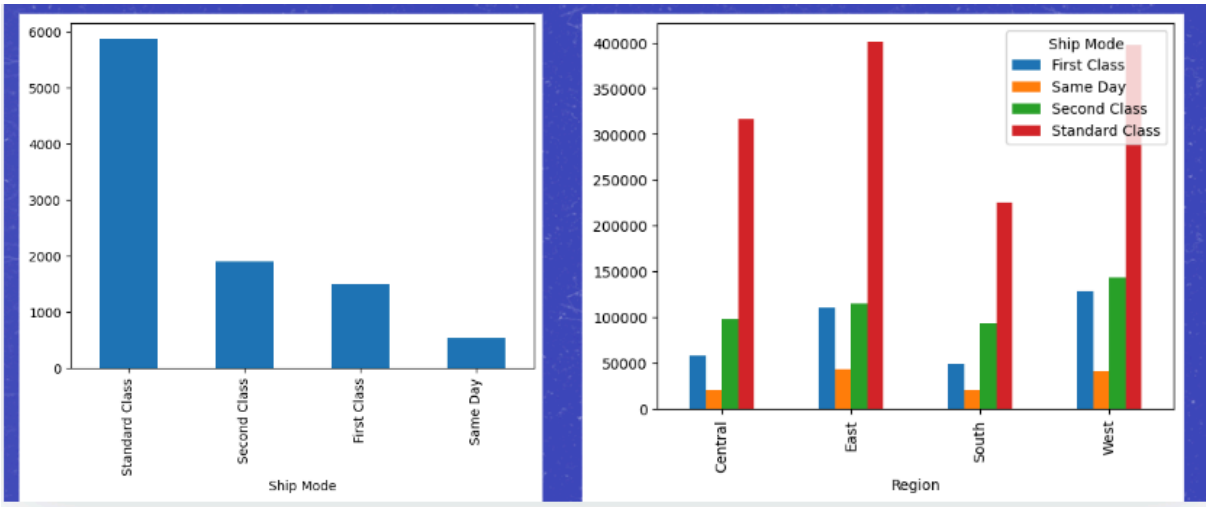

Monthly Sales Over Years
Year
2015
2016
2017
2018
120000
100000
80000
60000
40000
20000
0
Total Sales
1
2
3
4
5
6
7
8
9
10
11
12
Month

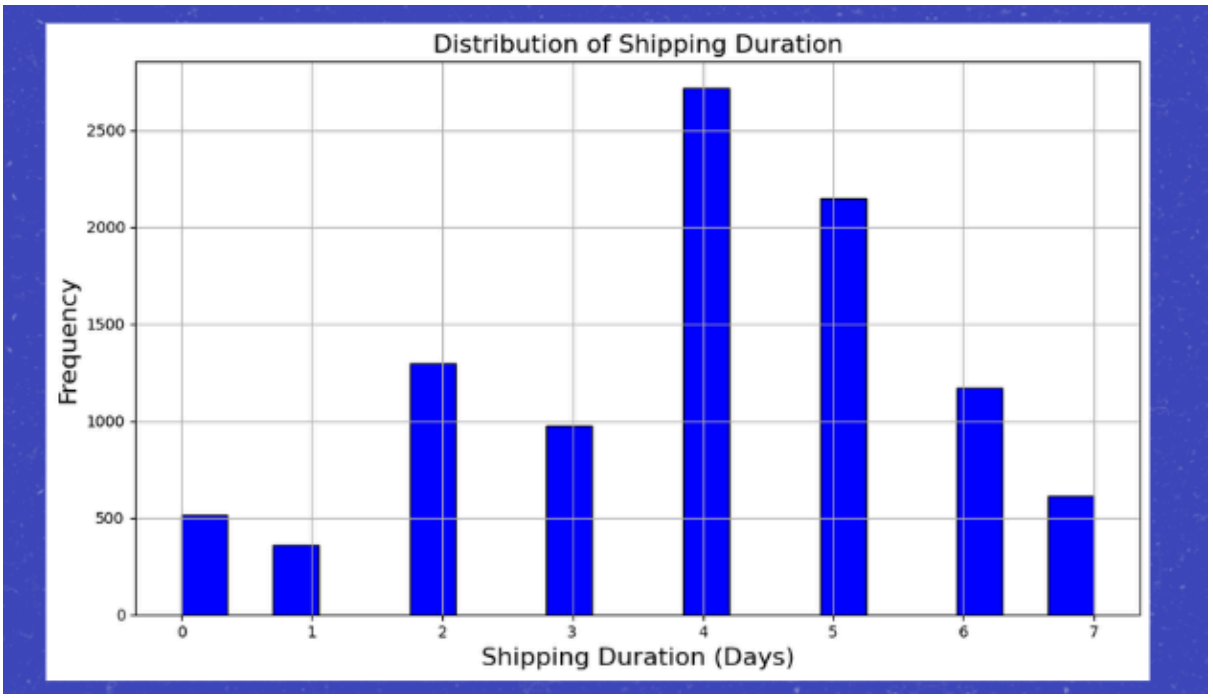

Total Sales Over Years
Total Sales
700000
650000
600000
550000
500000
450000
2015
2016
2017
2018
Year

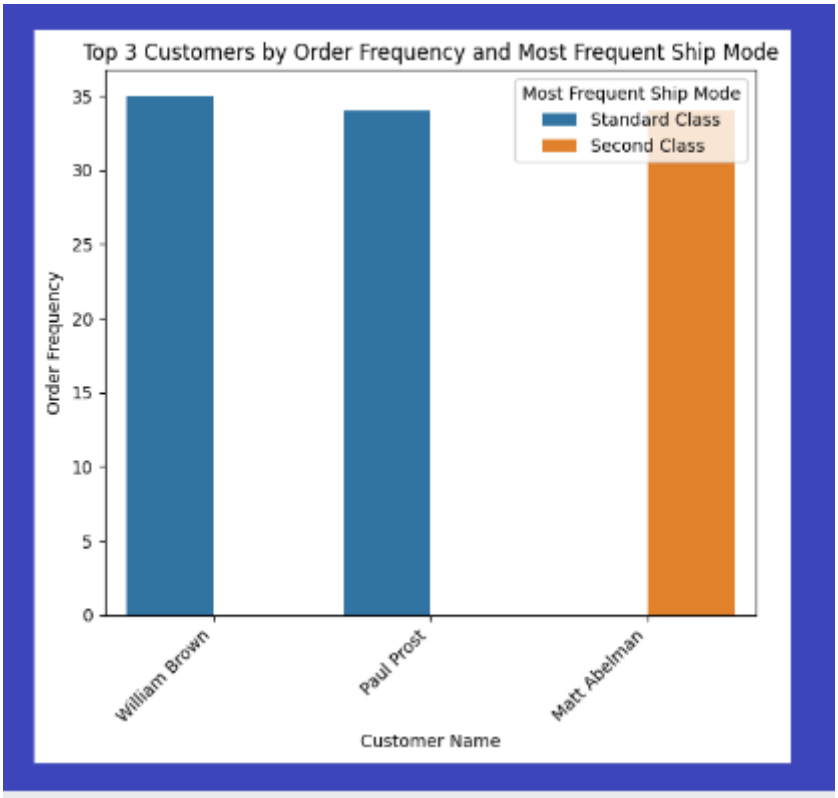

Monthly Sales in 2018
Total Sales
120000
100000
80000
60000
40000
20000
1
2
3
4
5
6
7
8
9
10
11
12
Month


Monthly Sales in 2016
Total Sales
Month
70000
60000
50000
40000
30000
20000
10000
1
2
3
4
5
6
7
8
9
10
11
12


Total Sales by Category per Year
Total Sales
Year
Category
Furniture
Office Supplies
Technology
260000
240000
220000
200000
180000
160000
140000
2015.0
2015.5
2016.0
2016.5
2017.0
2017.5
2018.0


6000
5000
4000
3000
2000
1000
0
Standard Class
Second Class
First Class
Same Day
Ship Mode
400000
350000
300000
250000
200000
150000
100000
50000
0
Central
East
South
West
Region
Ship Mode
First Class
Same Day
Second Class
Standard Class


Distribution of Shipping Duration
Frequency
2500
2000
1500
1000
500
0
0
1
2
3
4
5
6
7
Shipping Duration (Days)



Conclusion

- The Best Selling in the Sub Category is Phones and Chairs with the most Sales (most needed in daily life), both accounting for 29% of the Total Sales. Labels and Fasteners generated the least sales.
- Customers frequently choosing Standard Class may indicate satisfaction with the balance of time and cost, even if it involves longer waits.
- Cost-effectiveness: Customers might prioritize lower shipping costs, especially for non-urgent technology products.
- Product Types: Technology products like phones, accessories, and machines may not require immediate delivery, making Standard Class a more economical and practical choice
- This conclusion aligns with the idea that customers balance shipping costs with acceptable delivery times when choosing shipping options.
- Florida represent top state in sales As its Age class range (18:38 Y)
- The Best Selling in the Sub Category is Phones and Chairs with the most revenue, both accounting for 29% of the Total Sales. Labels and Fasteners generated the least revenue.
- The most profitable sub-category is Table with a net loss of ($18,000). It’s important to note that loss-making sales like tables, Bookcases, and Supplies make up 16% of total sales.

- And the most ordered is headed on office supplies that 60% most ordered, which 26% of those are for binders.
- Tables displayed decent sales but operated at a loss. Additionally, both Bookcases and Supplies resulted in a negative profit for the company.
- Customers frequently choosing Standard Class may indicate satisfaction with the balance of time and cost, even if it involves longer waits.
- These observations raise concerns, suggesting potential issues with the cost structure or pricing strategy, especially for the Tables, Bookcases, and Supplies subcategories.

Recommendation

Insights on Improvement Opportunities:

- Targeted Promotions: For customers who frequently use Standard Class for non-urgent purchases could encourage them to choose quicker shipping options.
- Apply discounts and offers for permanent customers
- Improved Communication on Delivery Times
- Enhance marketing strategy
- Make a customer survey
- After selling service
- Ensure warehouse is close to all regions, open new ones close to others

Recommended survey to test customer satisfaction about ship duration:

Section 1: Shipping Duration

1. How satisfied are you with the shipping speed of your recent purchase?
    - Very satisfied
    - Satisfied
    - Neutral
    - Dissatisfied
    - Very dissatisfied
2. Was the estimated delivery time communicated clearly during checkout?
    - Yes, very clearly
    - Somewhat clearly
    - Neutral
    - Not clear
    - Not communicated at all
3. Did your order arrive within the expected delivery time?
    - Yes
    - No, it arrived earlier
    - No, it arrived late
4. How would you rate the time it took for your order to be processed and shipped?
    - Excellent

- Good
- Average
- Poor
- Very poor

5. What is the maximum delivery time you are willing to accept for standard shipping?
    - 1-2 days
    - 3-5 days
    - 6-8 days
    - 9-12 days
    - More than 12 days

Section 2: Shipping Cost

6. How satisfied are you with the cost of shipping for your recent purchase?
    - Very satisfied
    - Satisfied
    - Neutral
    - Dissatisfied
    - Very dissatisfied
7. Do you feel that the shipping cost was fair for the speed of delivery?
    - Yes, it was fair
    - It was too expensive
    - It was cheaper than expected
    - I received free shipping
8. Would you be willing to pay more for faster shipping?
    - Yes, definitely
    - Maybe, depending on the cost
    - No, I prefer the current shipping speed
    - No, I would rather have free shipping even if it takes longer
9. What is the maximum amount you are willing to pay for faster (express) shipping?
    - Less than $5
    - $5 to $10
    - $10 to $15
    - $15 to $20
    - More than $20
10. How important is free shipping to your purchasing decision?
    - Extremely important
    - Very important
    - Somewhat important
    - Not important at all

Section 3: Overall Experience

11. How would you rate your overall satisfaction with the shipping experience (cost, speed, communication)?
    - Excellent

    - Good
    - Average
    - Poor
    - Very poor
12. How likely are you to shop again with this retailer based on your recent shipping experience?
    - Very likely
    - Somewhat likely
    - Neutral
    - Somewhat unlikely
    - Very unlikely
13. Did you encounter any issues with the tracking information provided during shipping?
    - No issues, everything was clear
    - Some delays in updating tracking
    - Significant problems with tracking
    - No tracking information provided
14. Which of the following factors would most likely influence your choice of shipping method in the future? (Select all that apply)
    - Cost of shipping
    - Speed of delivery
    - Delivery tracking
    - Reputation of the shipping carrier
    - Shipping insurance
15. Is there anything else you would like to share about your shipping experience?

Code

- Code is written a coordination between team members using pycharm and google colab
- Data frame name is df in colab code and superstore in pycharm

```
[ ]  # import required libraries
     import numpy as np
     import pandas as pd # data processing, CSV file
     import statistics # for calculations
     import matplotlib.pyplot as plt #for pie chart
     import seaborn as sns #for heatmap
     import plotly.express as px #for bar chart
     # Read the Excel file.
     df = pd.read_csv("//content/Superstore Sales Dataset.csv")
```

```
[ ]  df.shape
```

```
(9800, 18)
```

```
[ ]  #Explore data
     print (df.head())
     df.shape
     df.describe()
     df.info()
```

```
# Data cleaning
df.isnull()
df.isnull().sum()
#Only postal code has 11 missing values
```

```
#Replace null values with postal code of Burlington
df[df["Postal Code"].isnull()]
df["Postal Code"]= df["Postal Code"].fillna(05401.0)
df.isnull().sum()
```

```
#Change order date data type into date type
superstore['Order Date'] = pd.to_datetime(superstore['Order Date'], format='%d/%m/%Y')# Added format argument to specify day/month/year format
superstore['day'] = superstore['Order Date'].dt.day
superstore['month'] = superstore['Order Date'].dt.month
superstore['year'] = superstore['Order Date'].dt.year

#Change order date data type into date type
superstore['Ship Date'] = pd.to_datetime(superstore['Ship Date'], format='%d/%m/%Y')
superstore['day_ship'] = superstore['Ship Date'].dt.day
superstore['month_ship'] = superstore['Ship Date'].dt.month
superstore['year_ship'] = superstore['Ship Date'].dt.year
nan_superstore = superstore[superstore.isna().any(axis=1)]
```

```
[ ]  #What the trend of sales over years?
     yearly_sales = df.groupby('year')['Sales'].sum().reset_index()
     plt.figure(figsize=(10, 6))
     plt.plot(yearly_sales['year'], yearly_sales['Sales'], marker='o')  # Use plot for line plot
     plt.title('Trend of Sales Over Years')
     plt.xlabel('Year')
     plt.ylabel('Total Sales')
     plt.xticks(yearly_sales['year'])  # Set x-axis ticks to the years
     plt.grid(True)
     plt.tight_layout()
     plt.show()
```

```
[ ]  #Monthly sales over years
     monthly_sales_over_years = df.groupby(['year', 'month'])['Sales'].sum().reset_index()
     plt.figure(figsize=(12, 6))
     for year in monthly_sales_over_years['year'].unique():
         year_data = monthly_sales_over_years[monthly_sales_over_years['year'] == year]
         plt.plot(year_data['month'], year_data['Sales'], marker='o', label=year)
     plt.title('Monthly Sales Over Years')
     plt.xlabel('Month')
     plt.ylabel('Total Sales')
     plt.xticks(range(1, 13))
     plt.legend(title='Year')
     plt.grid(True)
     plt.tight_layout()
     plt.show()
```

```
#Monthly sales in 2016 as declined year
df_2016 = df[df['year'] == 2016]
monthly_sales_2016 = df_2016.groupby('month')['Sales'].sum().reset_index()
plt.figure(figsize=(10, 6))
plt.plot(monthly_sales_2016['month'], monthly_sales_2016['Sales'], marker='o')
plt.title('Monthly Sales in 2016')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.xticks(monthly_sales_2016['month'])
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
[ ]  #Monthly sales in 2018 as growth year
     # Filter data for 2018
     df_2018 = df[df['year'] == 2018]
     monthly_sales_2018 = df_2018.groupby('month')['Sales'].sum().reset_index()
     plt.figure(figsize=(10, 6))
     plt.plot(monthly_sales_2018['month'], monthly_sales_2018['Sales'], marker='o')
     plt.title('Monthly Sales in 2018')
     plt.xlabel('Month')
     plt.ylabel('Total Sales')
     plt.xticks(monthly_sales_2018['month'])
     plt.grid(True)
     plt.tight_layout()
     plt.show()
```

```
[ ]  # Calculate shipping duration (Ship Date - Order Date)
     df['Shipping Duration (Days)'] = (df['Ship Date'] - df['Order Date']).dt.days
     # Display the updated df with the new Shipping Duration column
     print(df[['Order Date', 'Ship Date', 'Shipping Duration (Days)']].head())
```

```
[ ]  # Plot the distribution of shipping duration
     plt.figure(figsize=(10, 6))
     plt.hist(df['Shipping Duration (Days)'], bins=20, color='blue', edgecolor='black')
     plt.title('Distribution of Shipping Duration', fontsize=16)
     plt.xlabel('Shipping Duration (Days)', fontsize=16)
     plt.ylabel('Frequency', fontsize=16)
     plt.grid(True)
     plt.tight_layout()
     plt.show()
```

```
[ ]  #What ship mode is mostly preferred?
     df["Ship Mode"].value_counts().plot(kind="bar")
```

```
[ ]  df.groupby(["Ship Mode"])[["Sales"]].sum().sort_values("Sales")
```

```
df.groupby(["Ship Mode"])[["Sales"]].sum().sort_values("Sales")
```

```
df.groupby(["Category"])[["Sales"]].sum().sort_values("Sales")
```

```
#What is the most sold category?
category_sales = df.groupby(["Category"])[["Sales"]].sum().sort_values("Sales", ascending=False)
plt.figure(figsize=(10, 6))
plt.bar(category_sales.index, category_sales["Sales"])
plt.xlabel("Category")
plt.ylabel("Total Sales")
plt.title("Total Sales by Each Category")
plt.xticks(rotation=45, ha="right")
plt.tight_layout()
```

```
#What is the most important segment in data?
df["Segment"].value_counts().plot(kind="pie",autopct="%1.1f%%")
```

```
#Total sales by segment
segment_sales = df.groupby(["Segment"])[["Sales"]].sum().sort_values("Sales", ascending=False)
plt.figure(figsize=(8, 6))
plt.bar(segment_sales.index, segment_sales["Sales"])
plt.xlabel("Segment")
plt.ylabel("Total Sales")
plt.title("Total Sales by Segment")
plt.tight_layout()
plt.show()
```

```
#What the sales of each category  in different segments?
segment_category_sales = pd.pivot_table(
    df,
    values='Sales',
    index='Segment',
    columns='Category',
    aggfunc='sum'
)

print(segment_category_sales)
```

```
segment_category_sales.plot(kind='bar', stacked=False)
plt.show()
```

```
#Sales in Regions
region_sales = df.groupby('Region')['Sales'].sum()
plt.figure(figsize=(10, 6))
plt.bar(region_sales.index, region_sales.values)
colors=
plt.xlabel("Region")
plt.ylabel("Total Sales")
plt.title("Total Sales by Region")
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
pivot_table = df.pivot_table(index='Region', columns='Ship Mode', values='Sales', aggfunc='sum')
pivot_table
```

```
pivot_table.plot(kind='bar', figsize=(10, 6))
plt.title('Total Sales by Region and Ship Mode')
plt.xlabel('Region')
plt.ylabel('Total Sales')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
```

```python
# Frequency: Number of orders per customer
frequency = df.groupby('Customer ID')['Order ID'].count().reset_index()
frequency.rename(columns={'Order ID': 'Frequency'}, inplace=True)

# Merge with customer names and ship mode
frequency = pd.merge(frequency, df[['Customer ID', 'Customer Name', 'Ship Mode']].drop_duplicates(), on='Customer ID', how='left')

# Get the most frequent ship mode for each customer
customer_ship_mode = df.groupby(['Customer ID', 'Ship Mode'])['Order ID'].count().reset_index()
customer_ship_mode = customer_ship_mode.sort_values(by=['Customer ID', 'Order ID'], ascending=[True, False])
customer_ship_mode = customer_ship_mode.drop_duplicates(subset=['Customer ID'], keep='first')  # Keep only the most frequent ship mode
customer_ship_mode.rename(columns={'Order ID': 'Ship Mode Count', 'Ship Mode': 'Most Frequent Ship Mode'}, inplace=True)

# Merge frequency with customer ship mode information
frequency = pd.merge(frequency, customer_ship_mode[['Customer ID', 'Most Frequent Ship Mode']], on='Customer ID', how='left')

# Sort by frequency and get the top 10
top_10_customers = frequency.sort_values(by=['Frequency'], ascending=False).head(10)

# Display the top 10 customer names, frequency, and most frequent ship mode
print(top_10_customers[['Customer Name', 'Frequency', 'Most Frequent Ship Mode']])
```

```python
# Frequency: Number of orders per customer
frequency = df.groupby('Customer ID')['Order ID'].count().reset_index()
frequency.rename(columns={'Order ID': 'Frequency'}, inplace=True)
frequency = pd.merge(frequency, df[['Customer ID', 'Customer Name']].drop_duplicates(), on='Customer ID', how='left')
# Sort by frequency and get the top 10
top_10_customers = frequency.sort_values(by=['Frequency'], ascending=False).head(10)
# Display the top 10 customer names
print(top_10_customers[['Customer Name', 'Frequency']])
top_10_customers.plot(kind='bar', x='Customer Name', y='Frequency', title='Top 10 Customers by Order Frequency')
plt.show()
```

```python
#Plotting the top 3 customers using a bar plot
plt.figure(figsize=(6, 6))
sns.barplot(x='Customer Name', y='Frequency', hue='Most Frequent Ship Mode', data=top_10_customers, dodge=True)
plt.xlabel("Customer Name")
plt.ylabel("Order Frequency")
plt.title("Top 3 Customers by Order Frequency and Most Frequent Ship Mode")
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.legend(title='Most Frequent Ship Mode', loc='upper right')
plt.show()
```

```python
#What the least 10 sub-categories in sales?
subcategory_sales = df.groupby('Sub-Category')['Sales'].sum().reset_index()

# Sort by sales in ascending order and get the top 10
least_10_subcategories = subcategory_sales.sort_values(by=['Sales']).head(10)

# Display the least 10 subcategories and their sales
print(least_10_subcategories)
plt.figure(figsize=(12, 6))
plt.bar(least_10_subcategories['Sub-Category'], least_10_subcategories['Sales'])
plt.title('10 Least Subcategories in Sales')
plt.xlabel('Sub-Category')
plt.ylabel('Total Sales')
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for better readability
plt.tight_layout()
plt.show()
```

```python
# Extract year from 'Order Date'
superstore['Year'] = superstore['Order Date'].dt.year

# Group by year and category, then sum sales
total_sales_by_category_per_year = superstore.groupby(['Year', 'Category'])['Sales'].sum().unstack()

# Plot total sales with main categories per year
total_sales_by_category_per_year.plot(kind='line', stacked=False)
plt.xlabel('Year')
plt.ylabel('Total Sales')
plt.title('Total Sales by Category per Year')
plt.legend(title='Category', loc='upper left', bbox_to_anchor=(1.05, 1))  # Adjust legend position
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.tight_layout()
plt.show()
```

```python
# Choose a specific year (replace with your desired year)
year_to_analyze = 2015

# Filter data for the chosen year
filtered_data = superstore[superstore['Order Date'].dt.year == year_to_analyze]

# Group by subcategory and month, then calculate sales
subcategory_sales_by_month = (
    filtered_data.groupby(["Sub-Category", filtered_data['Order Date'].dt.month])["Sales"].sum()
)

# Unstack the DataFrame to have months as columns
subcategory_sales_by_month = subcategory_sales_by_month.unstack()

# Plot line chart for each subcategory
plt.figure(figsize=(12, 6))
for subcategory in subcategory_sales_by_month.columns:
    plt.plot(subcategory_sales_by_month.index, subcategory_sales_by_month[subcategory], label=subcategory)

plt.xlabel("Month")
plt.ylabel("Total Sales")
plt.title(f"Monthly Sales for Subcategories in {year_to_analyze}")
plt.legend()
plt.show()
```

```python
# Filter data for the chosen category
filtered_data = superstore[superstore["Category"] == main_category]

# Group by subcategory and calculate total sales
subcategory_sales = filtered_data.groupby("Sub-Category")["Sales"].sum()

# Plot top-selling subcategories (adjust number of subcategories)
plt.figure(figsize=(10, 6))
top_subcategories = subcategory_sales.nlargest(5)  # Choose number of subcategories
plt.bar(top_subcategories.index, top_subcategories.values)
plt.xlabel("Sub-Category")
plt.ylabel("Total Sales")
plt.title(f"Top-Selling Subcategories in {main_category}")
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.tight_layout()
plt.show()
```