Superstore Analysis Documentation

TEAM MEMBERS:

MOHAMED NABIH
ADNAN OMAR
FARIS HAMADA
FATMA TAMER
ARWA ABAZA

Table of Contents

1. Project Overview

The **Superstore Analysis** project focuses on analyzing sales data from a retail chain to extract valuable insights that enhance decision-making. The analysis emphasizes key areas, including sales trends, customer behavior, product performance, and geographical sales distribution. Python is utilized for data processing, visualization, and storytelling.

2. Purpose

To enable the business to understand its performance across various segments, regions, and categories, facilitating data-driven decisions that enhance profitability and operational efficiency.

Key Questions

1. Which product categories generate the most revenue?

2. How do different regions perform in terms of sales and profit?

3. What customer segments are the most profitable?

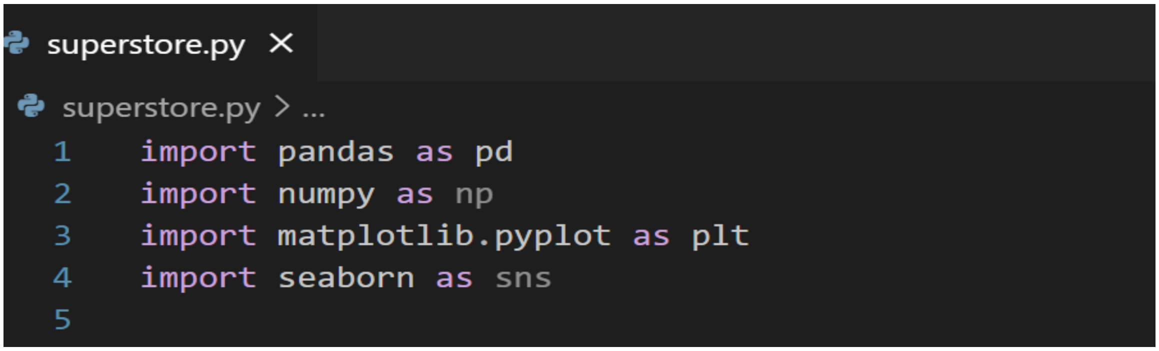
4. Are there seasonal trends affecting sales?

5. Which product subcategories require attention due to low performance?

3. Data Preprocessing

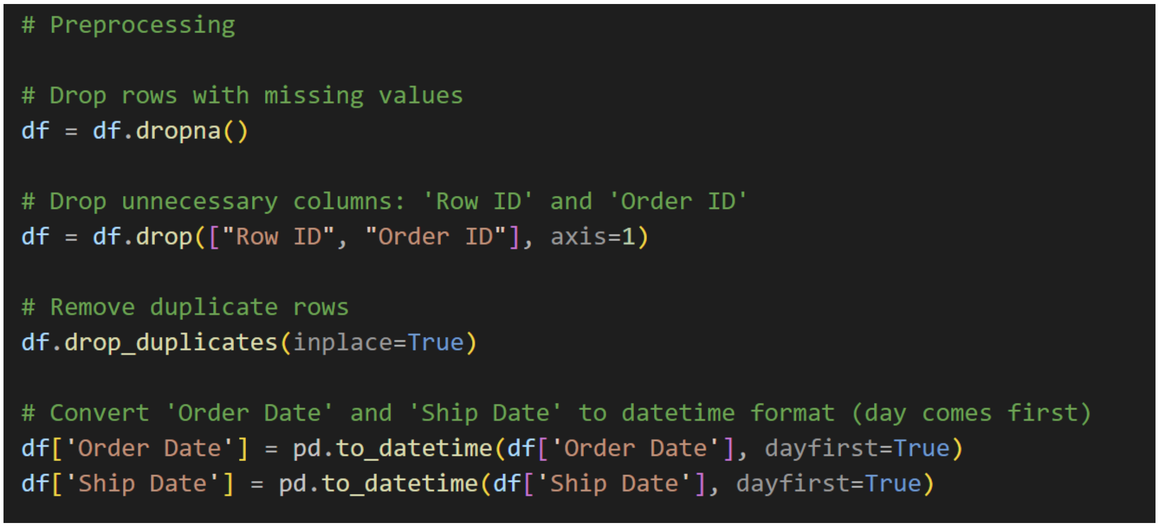
Steps

1. **Importing Libraries:** Load the necessary Python libraries for data manipulation and visualization. :



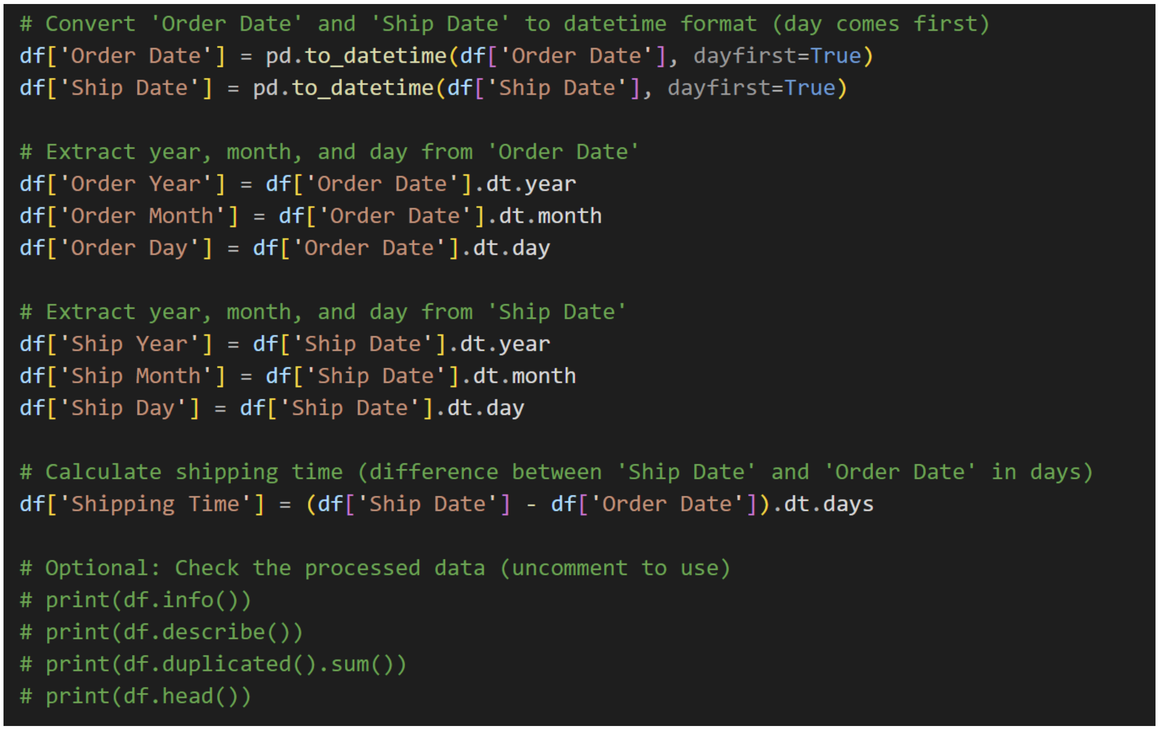
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

```

2. **Loading the Data:** Read the dataset into a pandas DataFrame.
3. **Data Cleaning:** Eliminate duplicates, correct data types, and clean any inconsistencies.



```
# Preprocessing

# Drop rows with missing values
df = df.dropna()

# Drop unnecessary columns: 'Row ID' and 'Order ID'
df = df.drop(["Row ID", "Order ID"], axis=1)

# Remove duplicate rows
df.drop_duplicates(inplace=True)

# Convert 'Order Date' and 'Ship Date' to datetime format (day comes first)
df['Order Date'] = pd.to_datetime(df['Order Date'], dayfirst=True)
df['Ship Date'] = pd.to_datetime(df['Ship Date'], dayfirst=True)
```

4. **Feature Engineering:** Create new features as needed, such as Year, Month, and Profit Margin.



```
# Convert 'Order Date' and 'Ship Date' to datetime format (day comes first)
df['Order Date'] = pd.to_datetime(df['Order Date'], dayfirst=True)
df['Ship Date'] = pd.to_datetime(df['Ship Date'], dayfirst=True)

# Extract year, month, and day from 'Order Date'
df['Order Year'] = df['Order Date'].dt.year
df['Order Month'] = df['Order Date'].dt.month
df['Order Day'] = df['Order Date'].dt.day

# Extract year, month, and day from 'Ship Date'
df['Ship Year'] = df['Ship Date'].dt.year
df['Ship Month'] = df['Ship Date'].dt.month
df['Ship Day'] = df['Ship Date'].dt.day

# Calculate shipping time (difference between 'Ship Date' and 'Order Date' in days)
df['Shipping Time'] = (df['Ship Date'] - df['Order Date']).dt.days

# Optional: Check the processed data (uncomment to use)
# print(df.info())
# print(df.describe())
# print(df.duplicated().sum())
# print(df.head())
```

Dashboard 1: Product and Category Insights

Purpose:

The goal of this dashboard is to provide key insights into the sales performance of individual products and product categories. By analyzing the top-selling products and categories, we can better understand consumer preferences, identify the most profitable products, and guide inventory and marketing strategies.

Key Metrics:

1. **Top 10 Most Sold Products**: This metric highlights the 10 products with the highest sales volume, helping us identify the most popular products.
2. **Top 10 Products Generating the Most Sales**: This section focuses on the products that generate the most revenue, regardless of the number of units sold. It helps in identifying high-revenue products.
3. **Top Selling Product Categories**: Understanding which product categories contribute the most to sales can help optimize product mix and marketing efforts.

Analysis Methodology:

- **Top 10 Most Sold Products**:
  - The products were ranked based on the total number of units sold. We used the value_counts() method to count the occurrences of each product and selected the top 10.
- **Top 10 Products Generating the Most Sales**:
  - To identify the top revenue-generating products, we grouped the data by product name and summed the sales values. The products were then sorted by total sales, and the top 10 were selected.
- **Top Selling Product Categories**:
  - The sales data was grouped by category to show the overall contribution of each product category. The categories were ranked based on the total sales volume.

Visualizations:

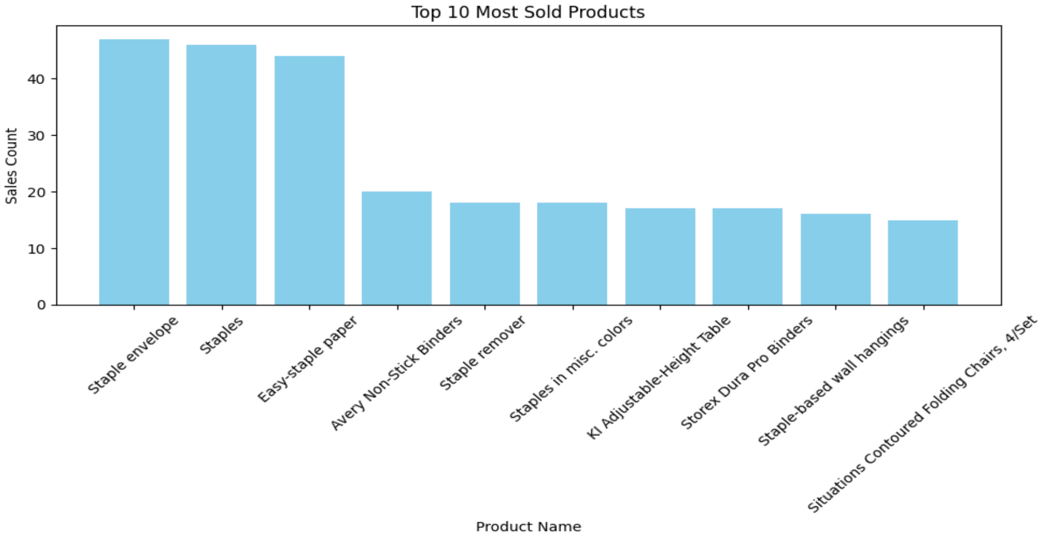
1. **Bar Chart for Top 10 Most Sold Products**:
   - A bar chart was used to represent the sales count of the top 10 most popular products. This visualization helps to easily compare the popularity of different products.
2. **Colored Bar Chart for Top 10 Revenue-Generating Products**:
   - A bar chart with distinct colors for each bar shows the top 10 products in terms of revenue. A legend is included to identify each product. This helps in visualizing which products generate the highest revenue.
3. **Pie Chart for Top Selling Product Categories**:
   - A pie chart was chosen to illustrate the percentage contribution of each product category to total sales. This chart provides a quick overview of which categories are performing well.

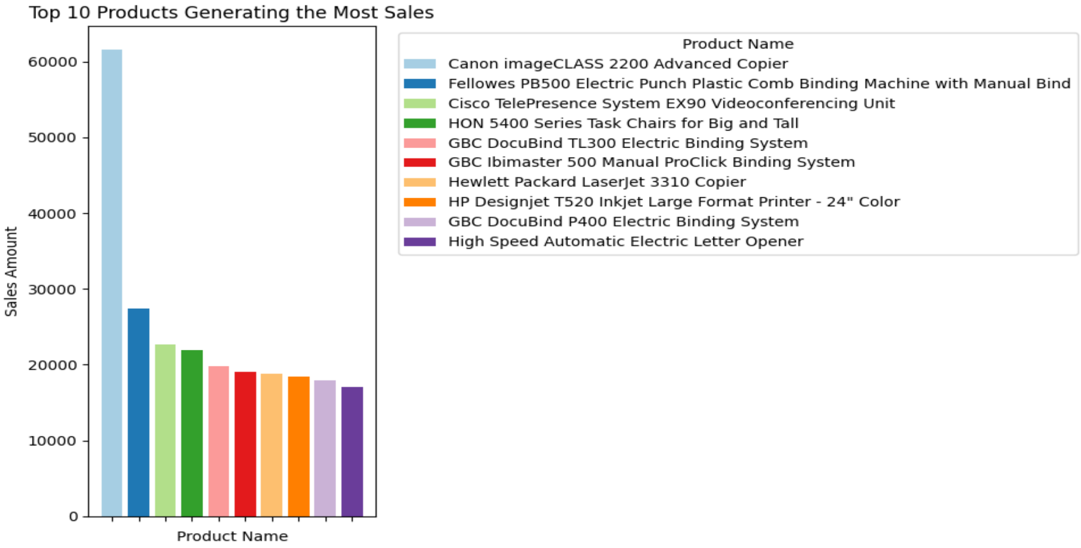
Insights:

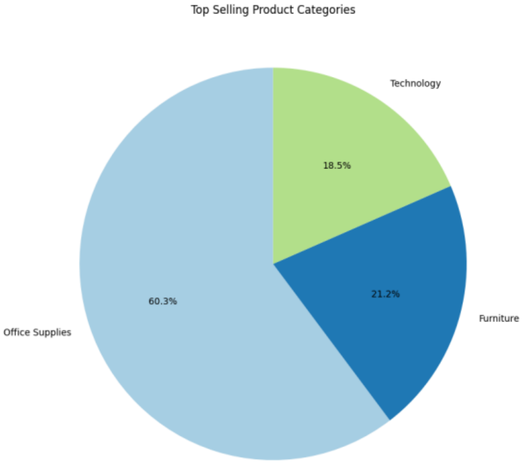
- The top-selling products give a clear indication of what consumers are most interested in, allowing the business to focus on stocking and promoting these items.
- The top revenue-generating products may differ from the most sold products, indicating that some high-value items contribute significantly to overall revenue, even if they are sold in smaller quantities.
- The category-level analysis helps in understanding which broader product types are most profitable and might require increased inventory, marketing focus, or pricing adjustments.

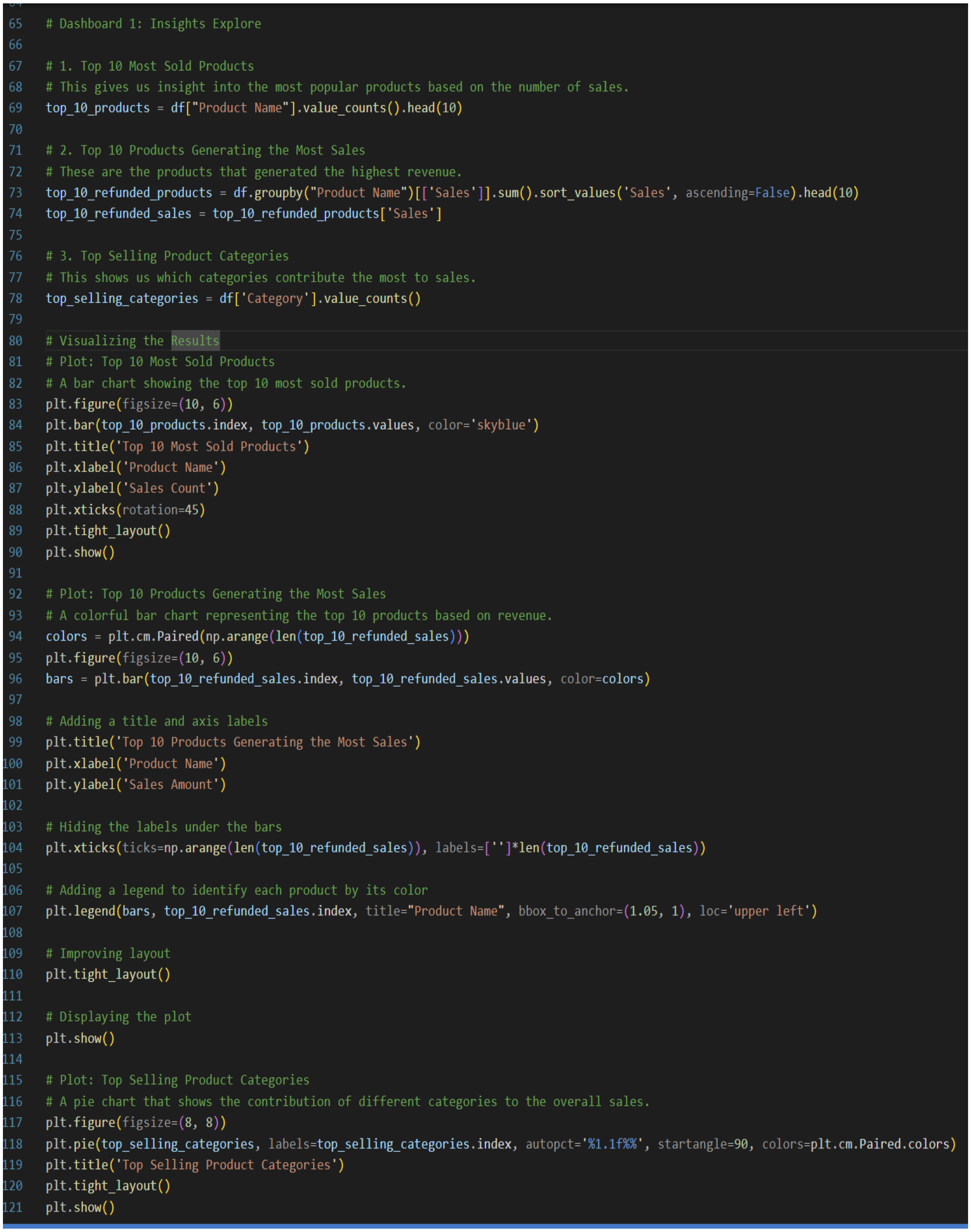
Business Impact:

- **Marketing**: Use this data to focus promotional campaigns on the most popular and high-revenue products.
- **Inventory Management**: Ensure the availability of top-selling products to avoid stockouts, while adjusting the stock for underperforming products.
- **Strategic Planning**: Align marketing, sales, and inventory efforts with the product categories that contribute the most to overall business performance.


Top 10 Most Sold Products
Sales Count
0
10
20
30
40
Staple envelope
Staples
Easy-staple paper
Avery Non-Stick Binders
Staple remover
Staples in misc. colors
KI Adjustable-Height Table
Storex Dura Pro Binders
Staple-based wall hangings
Situations Contoured Folding Chairs, 4/Set
Product Name


Top 10 Products Generating the Most Sales
Sales Amount
0
10000
20000
30000
40000
50000
60000
Product Name
Product Name
Canon imageCLASS 2200 Advanced Copier
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind
Cisco TelePresence System EX90 Videoconferencing Unit
HON 5400 Series Task Chairs for Big and Tall
GBC DocuBind TL300 Electric Binding System
GBC Ibimaster 500 Manual ProClick Binding System
Hewlett Packard LaserJet 3310 Copier
HP Designjet T520 Inkjet Large Format Printer - 24" Color
GBC DocuBind P400 Electric Binding System
High Speed Automatic Electric Letter Opener


Top Selling Product Categories
Technology
18.5%
Furniture
21.2%
Office Supplies
60.3%



```python
# Dashboard 1: Insights Explore

# 1. Top 10 Most Sold Products
# This gives us insight into the most popular products based on the number of sales.
top_10_products = df["Product Name"].value_counts().head(10)

# 2. Top 10 Products Generating the Most Sales
# These are the products that generated the highest revenue.
top_10_refunded_products = df.groupby("Product Name")[['Sales']].sum().sort_values('Sales', ascending=False).head(10)
top_10_refunded_sales = top_10_refunded_products['Sales']

# 3. Top Selling Product Categories
# This shows us which categories contribute the most to sales.
top_selling_categories = df['Category'].value_counts()

# Visualizing the Results
# Plot: Top 10 Most Sold Products
# A bar chart showing the top 10 most sold products.
plt.figure(figsize=(10, 6))
plt.bar(top_10_products.index, top_10_products.values, color='skyblue')
plt.title('Top 10 Most Sold Products')
plt.xlabel('Product Name')
plt.ylabel('Sales Count')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Plot: Top 10 Products Generating the Most Sales
# A colorful bar chart representing the top 10 products based on revenue.
colors = plt.cm.Paired(np.arange(len(top_10_refunded_sales)))
plt.figure(figsize=(10, 6))
bars = plt.bar(top_10_refunded_sales.index, top_10_refunded_sales.values, color=colors)

# Adding a title and axis labels
plt.title('Top 10 Products Generating the Most Sales')
plt.xlabel('Product Name')
plt.ylabel('Sales Amount')

# Hiding the labels under the bars
plt.xticks(ticks=np.arange(len(top_10_refunded_sales)), labels=['']*len(top_10_refunded_sales))

# Adding a legend to identify each product by its color
plt.legend(bars, top_10_refunded_sales.index, title="Product Name", bbox_to_anchor=(1.05, 1), loc='upper left')

# Improving layout
plt.tight_layout()

# Displaying the plot
plt.show()

# Plot: Top Selling Product Categories
# A pie chart that shows the contribution of different categories to the overall sales.
plt.figure(figsize=(8, 8))
plt.pie(top_selling_categories, labels=top_selling_categories.index, autopct='%1.1f%%', startangle=90, colors=plt.cm.Paired.colors)
plt.title('Top Selling Product Categories')
plt.tight_layout()
plt.show()
```

Dashboards 2: Shipping, Regional, and Sales Time Analysis

Purpose:

This dashboard is designed to provide insights into the performance of shipping methods, regional sales distribution, and sales trends over time. It focuses on the shipping modes used by customers, how sales are distributed across regions, and the sales performance of different time periods, including monthly and yearly trends.

Key Metrics:

1. **Most Used Shipping Method**: Understanding the preferred shipping methods can help optimize logistics and improve customer satisfaction.
2. **Monthly Sales Trends Over the Years**: This provides insight into how sales fluctuate across months and years, helping to identify seasonality or growth trends.
3. **Average Shipping Time by Year**: Monitoring the average shipping time is critical for improving customer experience and logistics efficiency.
4. **Sales by Region**: Identifying which regions generate the most sales can help prioritize regional marketing and sales efforts.
5. **Top Cities by Sales**: This metric identifies which cities are driving the highest revenue, which can help in regional strategy.
6. **Top States by Sales**: Like cities, identifying top-performing states will provide insight into larger-scale regional performance.

Analysis Methodology:

- **Most Used Shipping Method**:
  - The shipping methods are categorized, and the usage frequency is calculated using the value_counts() method. A pie chart is used to visualize the distribution of different shipping modes.
- **Monthly Sales Trends Over the Years**:
  - Sales data is grouped by the year and month, and a line plot is used to show trends over time. This visualization helps identify which months perform the best across different years.
- **Average Shipping Time by Year**:

  - The average shipping time is calculated for each year to assess performance over time. This information is then plotted as a bar chart to compare shipping times year by year.

- **Sales by Region**:
  - The total sales for each region are calculated, and a bar chart is used to compare the regions in terms of sales volume. This allows a clear understanding of which regions perform better.

- **Top Cities by Sales**:
  - The sales data is grouped by city, and the top 10 cities by sales are selected. A bar chart is used to visualize the top-performing cities.

- **Top States by Sales**:
  - Similar to cities, the data is grouped by state, and the top 10 states by sales are identified and visualized in a bar chart.

Visualizations:

1. **Pie Chart for Most Used Shipping Method**:
   - A pie chart is used to visualize the percentage usage of different shipping methods. This helps in understanding customer preferences in shipping.
2. **Line Plot for Monthly Sales Trends**:
   - A multi-line plot shows the sales trends for each year, with the months on the x-axis and total sales on the y-axis. The lines are colored to differentiate between the years.
3. **Bar Chart for Average Shipping Time by Year**:
   - A bar chart is used to compare the average shipping time for each year. The chart provides an overview of how shipping times have evolved and whether there are any improvements.
4. **Bar Chart for Sales by Region**:
   - A horizontal bar chart helps visualize the total sales across different regions, allowing for easy comparison between regions.
5. **Bar Chart for Top Cities by Sales**:
   - A vertical bar chart shows the top 10 cities by sales, giving insights into urban areas contributing the most to revenue.

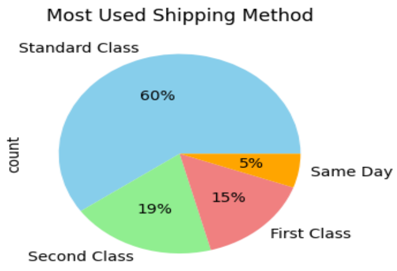
6. **Bar Chart for Top States by Sales**:
   - Another vertical bar chart shows the top 10 states by sales, providing a broader regional view of performance.

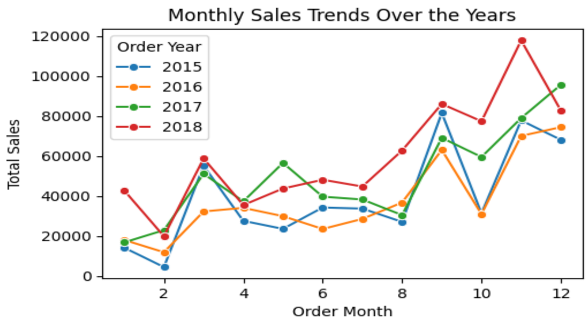
Insights:

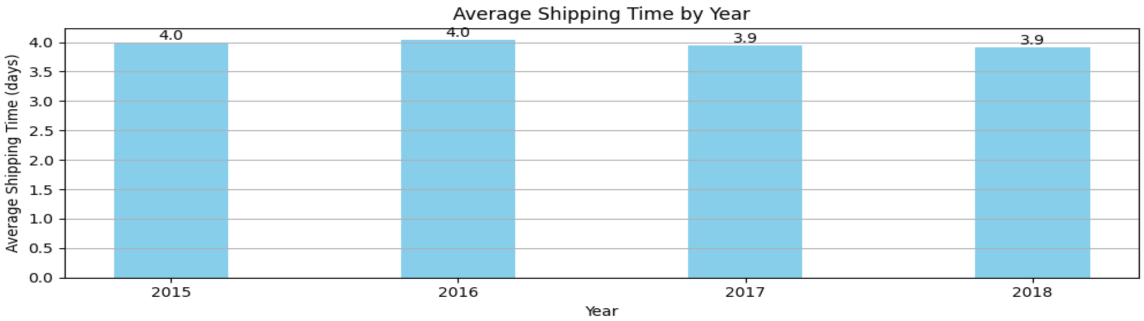
- **Shipping Methods**: The analysis of shipping methods shows the most preferred options, helping the business tailor its logistics approach to customer preferences.
- **Monthly and Yearly Sales Trends**: The trends show which months typically have higher or lower sales, which can help in planning promotions, inventory, and staffing.
- **Shipping Time**: Monitoring shipping time over the years helps evaluate logistics efficiency and identify areas for improvement.
- **Regional Sales Performance**: The regional breakdown highlights which areas should be prioritized for sales and marketing efforts, while understanding high-performing cities and states can guide more localized strategies.

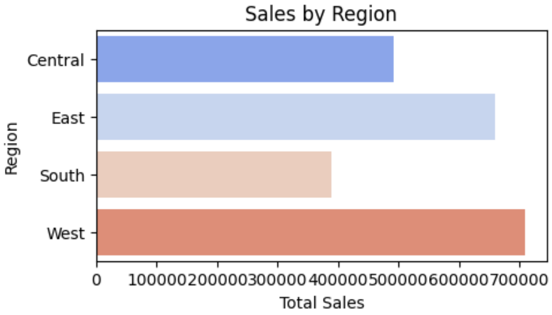
Business Impact:

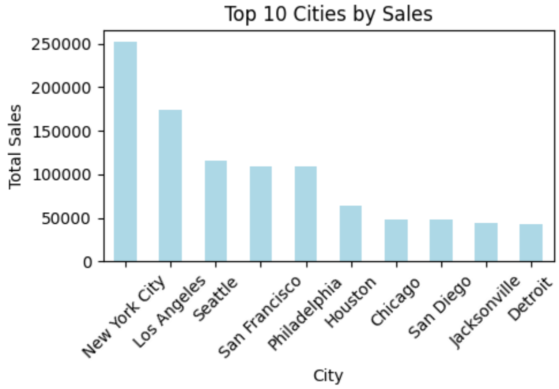
- **Logistics Optimization**: Understanding the preferred shipping methods and tracking shipping times can help the business improve its logistics strategy.
- **Seasonal Sales Planning**: Insights into monthly sales trends allow for better preparation and resource allocation during high-demand periods.
- **Regional Strategy**: Focusing on top-performing regions, cities, and states can help drive more targeted marketing campaigns and improve local sales performance.

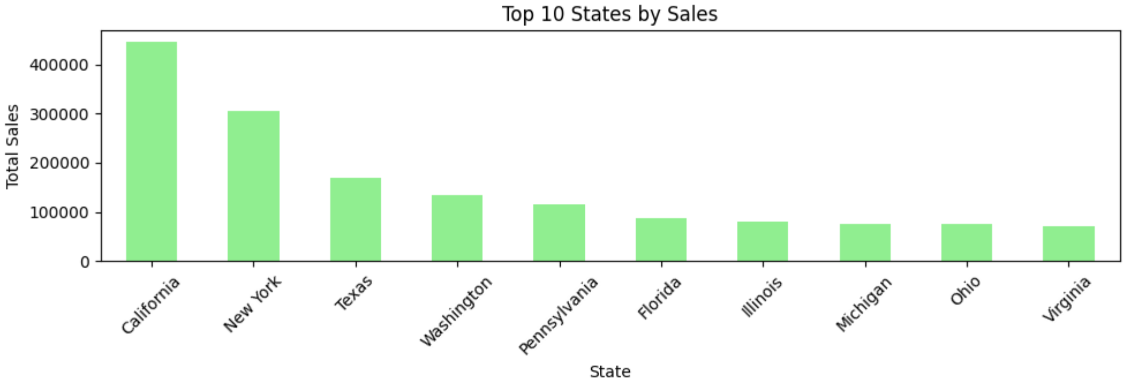

Most Used Shipping Method

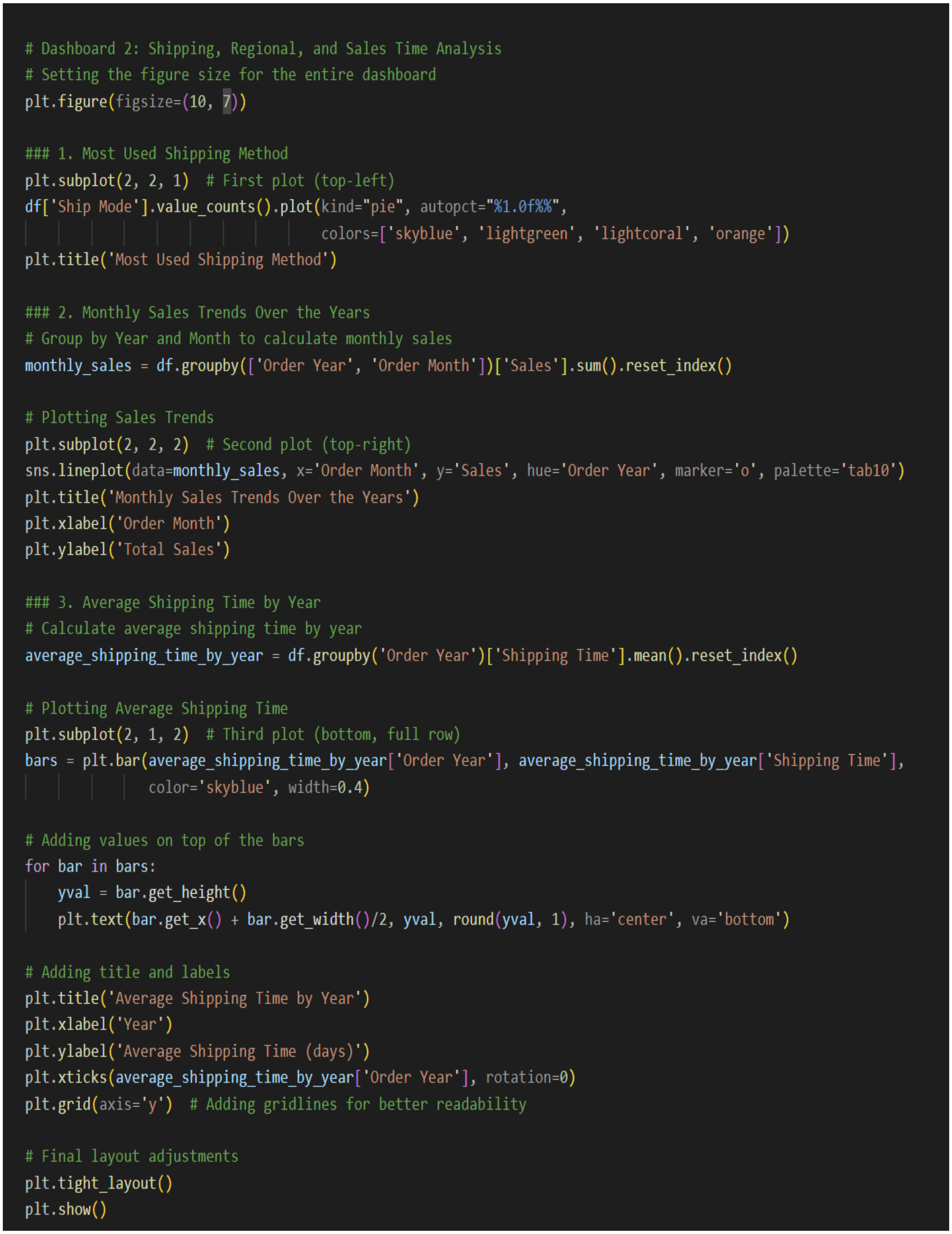

Monthly Sales Trends Over the Years


Average Shipping Time by Year

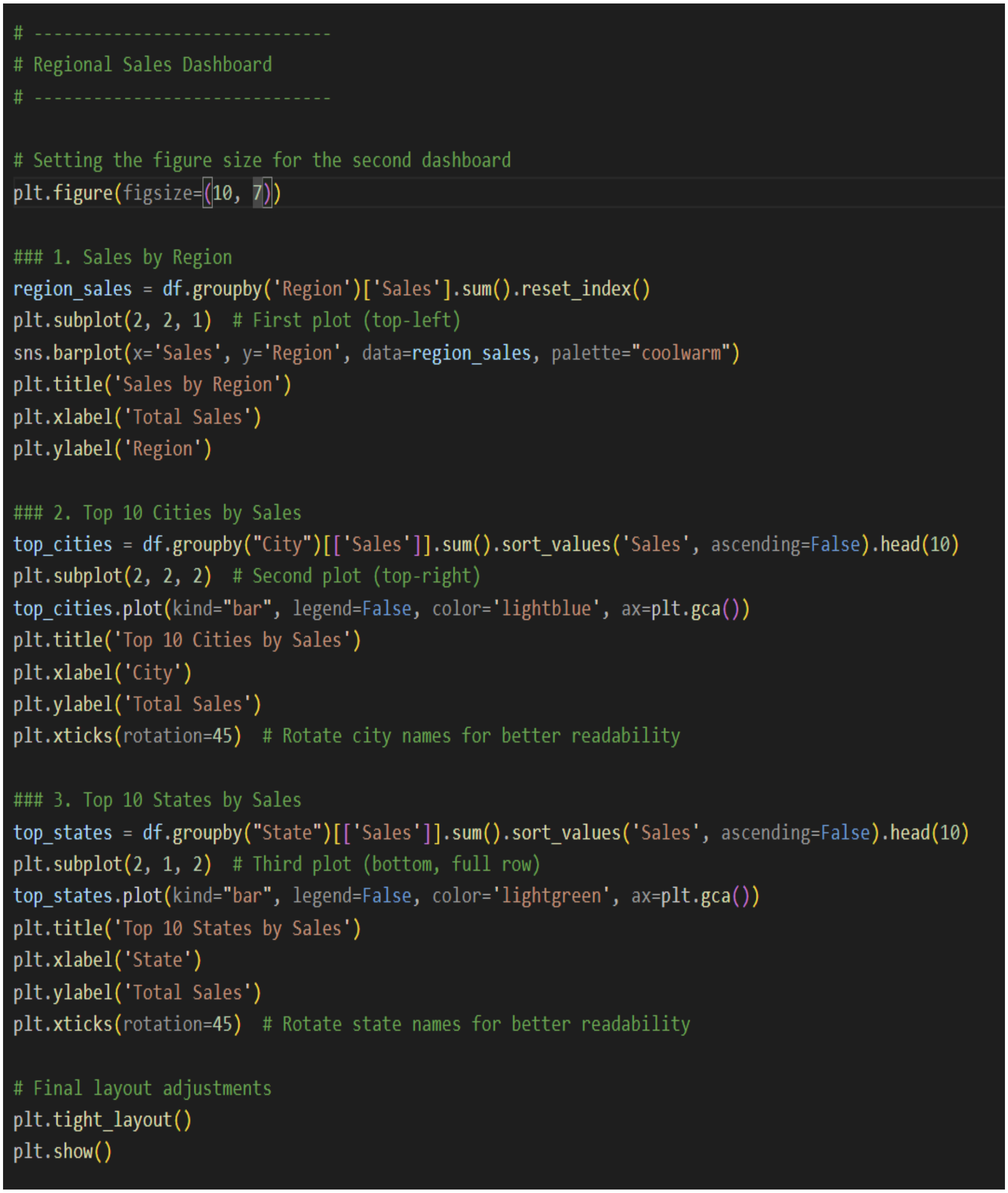

Sales by Region


Top 10 Cities by Sales


Top 10 States by Sales



```
# Dashboard 2: Shipping, Regional, and Sales Time Analysis
# Setting the figure size for the entire dashboard
plt.figure(figsize=(10, 7))

### 1. Most Used Shipping Method
plt.subplot(2, 2, 1)  # First plot (top-left)
df['Ship Mode'].value_counts().plot(kind="pie", autopct="%1.0f%%",
                                    colors=['skyblue', 'lightgreen', 'lightcoral', 'orange'])
plt.title('Most Used Shipping Method')

### 2. Monthly Sales Trends Over the Years
# Group by Year and Month to calculate monthly sales
monthly_sales = df.groupby(['Order Year', 'Order Month'])['Sales'].sum().reset_index()

# Plotting Sales Trends
plt.subplot(2, 2, 2)  # Second plot (top-right)
sns.lineplot(data=monthly_sales, x='Order Month', y='Sales', hue='Order Year', marker='o', palette='tab10')
plt.title('Monthly Sales Trends Over the Years')
plt.xlabel('Order Month')
plt.ylabel('Total Sales')

### 3. Average Shipping Time by Year
# Calculate average shipping time by year
average_shipping_time_by_year = df.groupby('Order Year')['Shipping Time'].mean().reset_index()

# Plotting Average Shipping Time
plt.subplot(2, 1, 2)  # Third plot (bottom, full row)
bars = plt.bar(average_shipping_time_by_year['Order Year'], average_shipping_time_by_year['Shipping Time'],
               color='skyblue', width=0.4)

# Adding values on top of the bars
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval, round(yval, 1), ha='center', va='bottom')

# Adding title and labels
plt.title('Average Shipping Time by Year')
plt.xlabel('Year')
plt.ylabel('Average Shipping Time (days)')
plt.xticks(average_shipping_time_by_year['Order Year'], rotation=0)
plt.grid(axis='y')  # Adding gridlines for better readability

# Final layout adjustments
plt.tight_layout()
plt.show()
```



```python
# ------------------------------
# Regional Sales Dashboard
# ------------------------------

# Setting the figure size for the second dashboard
plt.figure(figsize=(10, 7))

### 1. Sales by Region
region_sales = df.groupby('Region')['Sales'].sum().reset_index()
plt.subplot(2, 2, 1)  # First plot (top-left)
sns.barplot(x='Sales', y='Region', data=region_sales, palette="coolwarm")
plt.title('Sales by Region')
plt.xlabel('Total Sales')
plt.ylabel('Region')

### 2. Top 10 Cities by Sales
top_cities = df.groupby("City")[['Sales']].sum().sort_values('Sales', ascending=False).head(10)
plt.subplot(2, 2, 2)  # Second plot (top-right)
top_cities.plot(kind="bar", legend=False, color='lightblue', ax=plt.gca())
plt.title('Top 10 Cities by Sales')
plt.xlabel('City')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)  # Rotate city names for better readability

### 3. Top 10 States by Sales
top_states = df.groupby("State")[['Sales']].sum().sort_values('Sales', ascending=False).head(10)
plt.subplot(2, 1, 2)  # Third plot (bottom, full row)
top_states.plot(kind="bar", legend=False, color='lightgreen', ax=plt.gca())
plt.title('Top 10 States by Sales')
plt.xlabel('State')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)  # Rotate state names for better readability

# Final layout adjustments
plt.tight_layout()
plt.show()
```

Dashboard 3: Customer Insights

Overview

The third dashboard provides insights into customer behavior by analyzing sales performance across different customer segments and identifying the top customers based on average sales. This information helps businesses tailor marketing strategies and improve customer relationship management.

Insights Explored

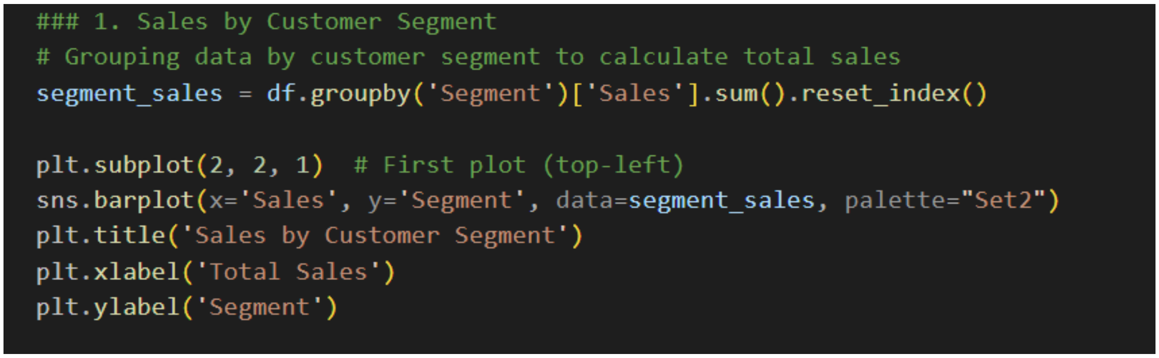
1. **Sales by Customer Segment**

   - **Description**: This chart illustrates the total sales generated from different customer segments, such as Consumer, Corporate, and Home Office.

   - **Purpose**: Understanding which segments contribute the most to sales can inform marketing strategies and product offerings.

   - **Recommended Chart**: Bar chart displaying total sales for each customer segment.

   - **Code Snippet**:



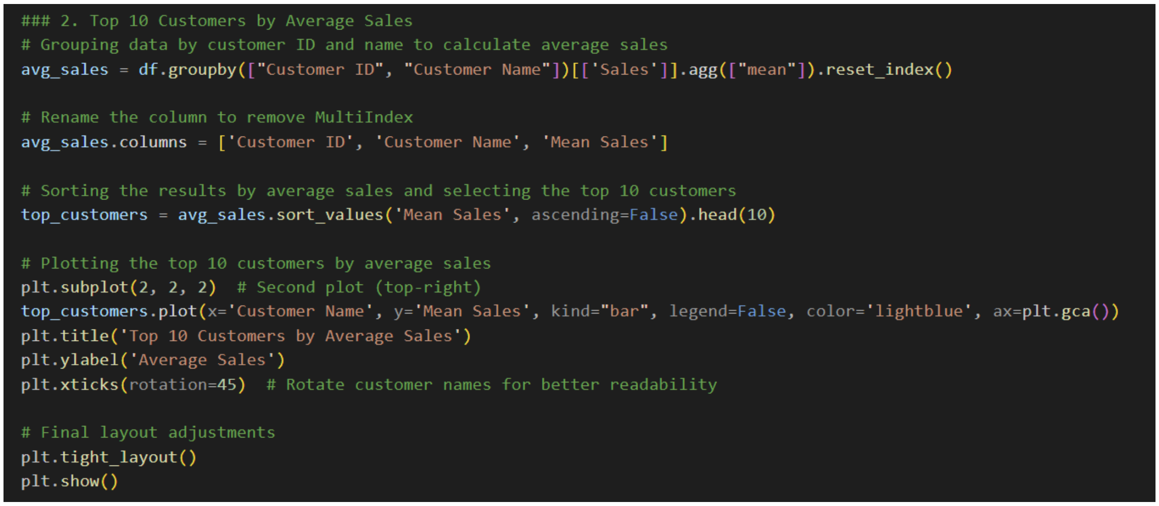
```
### 1. Sales by Customer Segment
# Grouping data by customer segment to calculate total sales
segment_sales = df.groupby('Segment')['Sales'].sum().reset_index()

plt.subplot(2, 2, 1)  # First plot (top-left)
sns.barplot(x='Sales', y='Segment', data=segment_sales, palette="Set2")
plt.title('Sales by Customer Segment')
plt.xlabel('Total Sales')
plt.ylabel('Segment')
```

2. **Top 10 Customers by Average Sales**

   - **Description**: This visualization identifies the top 10 customers based on their average sales. It highlights who the most valuable customers are in terms of purchase value.

   - **Purpose**: Recognizing top customers allows for targeted marketing efforts, such as personalized promotions or loyalty programs to retain these valuable customers.

- **Recommended Chart**: Bar chart showcasing the average sales of the top 10 customers.
- **Code Snippet**:



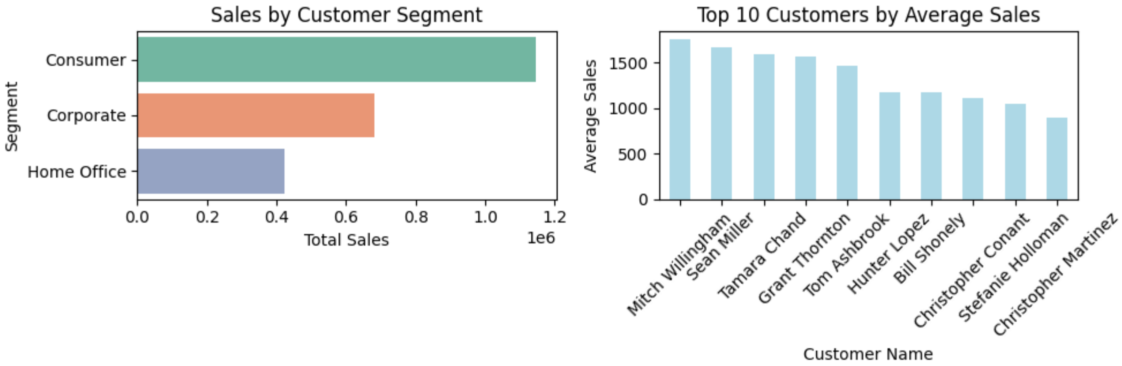
```
### 2. Top 10 Customers by Average Sales
# Grouping data by customer ID and name to calculate average sales
avg_sales = df.groupby(["Customer ID", "Customer Name"])[['Sales']].agg(["mean"]).reset_index()

# Rename the column to remove MultiIndex
avg_sales.columns = ['Customer ID', 'Customer Name', 'Mean Sales']

# Sorting the results by average sales and selecting the top 10 customers
top_customers = avg_sales.sort_values('Mean Sales', ascending=False).head(10)

# Plotting the top 10 customers by average sales
plt.subplot(2, 2, 2)  # Second plot (top-right)
top_customers.plot(x='Customer Name', y='Mean Sales', kind="bar", legend=False, color='lightblue', ax=plt.gca())
plt.title('Top 10 Customers by Average Sales')
plt.ylabel('Average Sales')
plt.xticks(rotation=45)  # Rotate customer names for better readability

# Final layout adjustments
plt.tight_layout()
plt.show()
```

Key Findings

- **Segment Performance**: The total sales by segment reveal which areas of the business are performing well, indicating where to focus sales efforts.
- **Customer Value**: Identifying top customers allows the business to build stronger relationships with them, potentially increasing customer loyalty and lifetime value.

Conclusion

This dashboard serves as a vital tool for understanding customer dynamics, enabling the business to leverage insights for strategic planning and decision-making. By focusing on segments and top customers, the organization can effectively allocate resources to maximize sales potential.


Sales by Customer Segment
Segment
Consumer
Corporate
Home Office
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1e6
Total Sales
Top 10 Customers by Average Sales
Average Sales
0
500
1000
1500
Mitch Willingham
Sean Miller
Tamara Chand
Grant Thornton
Tom Ashbrook
Hunter Lopez
Bill Shonely
Christopher Conant
Stefanie Holloman
Christopher Martinez
Customer Name

Dashboard 4: Shipping Time Analysis

Purpose

The purpose of this analysis is to evaluate delayed shipping orders to understand the contributing factors behind shipping delays. By identifying trends related to shipping methods, regions, and product categories, the analysis aims to provide actionable insights that can enhance operational efficiency and improve customer satisfaction.

Key Metrics

The following key metrics are utilized to assess shipping performance:

- **Average Shipping Time**: The mean number of days taken for orders to be shipped, categorized by shipping method, region, and product category.
- **Delayed Orders Count**: The total number of orders with shipping times exceeding 5 days.
- **Shipping Mode Distribution**: The breakdown of orders based on the shipping methods utilized.
- **Regional Performance**: Comparison of shipping times across different geographical areas.

Methodology

1. **Data Collection**: A pandas DataFrame named df is prepared, containing relevant columns such as Shipping Time, Ship Mode, Region, and Category.
2. **Filtering Delayed Orders**: Orders with a shipping time greater than 5 days are filtered to create a new DataFrame called delayed_orders.
3. **Calculating Averages**:
    - The average shipping time is computed for:
        - **Shipping Mode**: Grouped by Ship Mode.
        - **Region**: Grouped by Region.
        - **Product Category**: Grouped by Category.

4. **Visualization**: Bar charts are created to visualize the average shipping times based on the analyzed factors.

Visualization

The analysis results are presented through a dashboard containing three bar charts:

1. **Average Shipping Time by Ship Mode**
   - This chart displays the average number of days taken for shipping based on the method used.
2. **Average Shipping Time by Region**
   - This chart presents the average shipping times across various regions.
3. **Average Shipping Time by Product Category**
   - This chart shows the average shipping times for different product categories.

Each bar is annotated with the corresponding average shipping time to improve clarity.

Insights

The analysis provides several key insights:

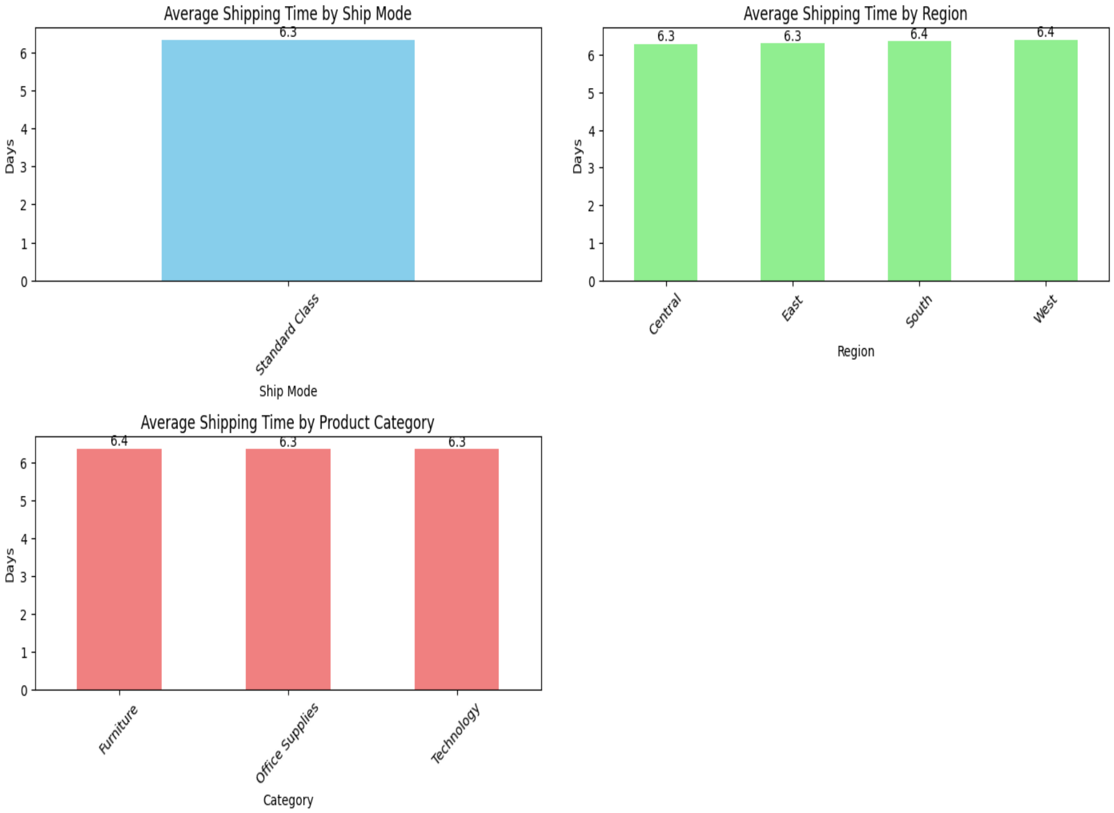
- **Shipping Mode Trends**: Identification of which shipping methods consistently result in longer shipping times.
- **Regional Performance**: Understanding which regions experience more delays, indicating potential logistical issues.
- **Product Category Analysis**: Insight into whether certain product categories are more prone to shipping delays, aiding in inventory and supply chain management.

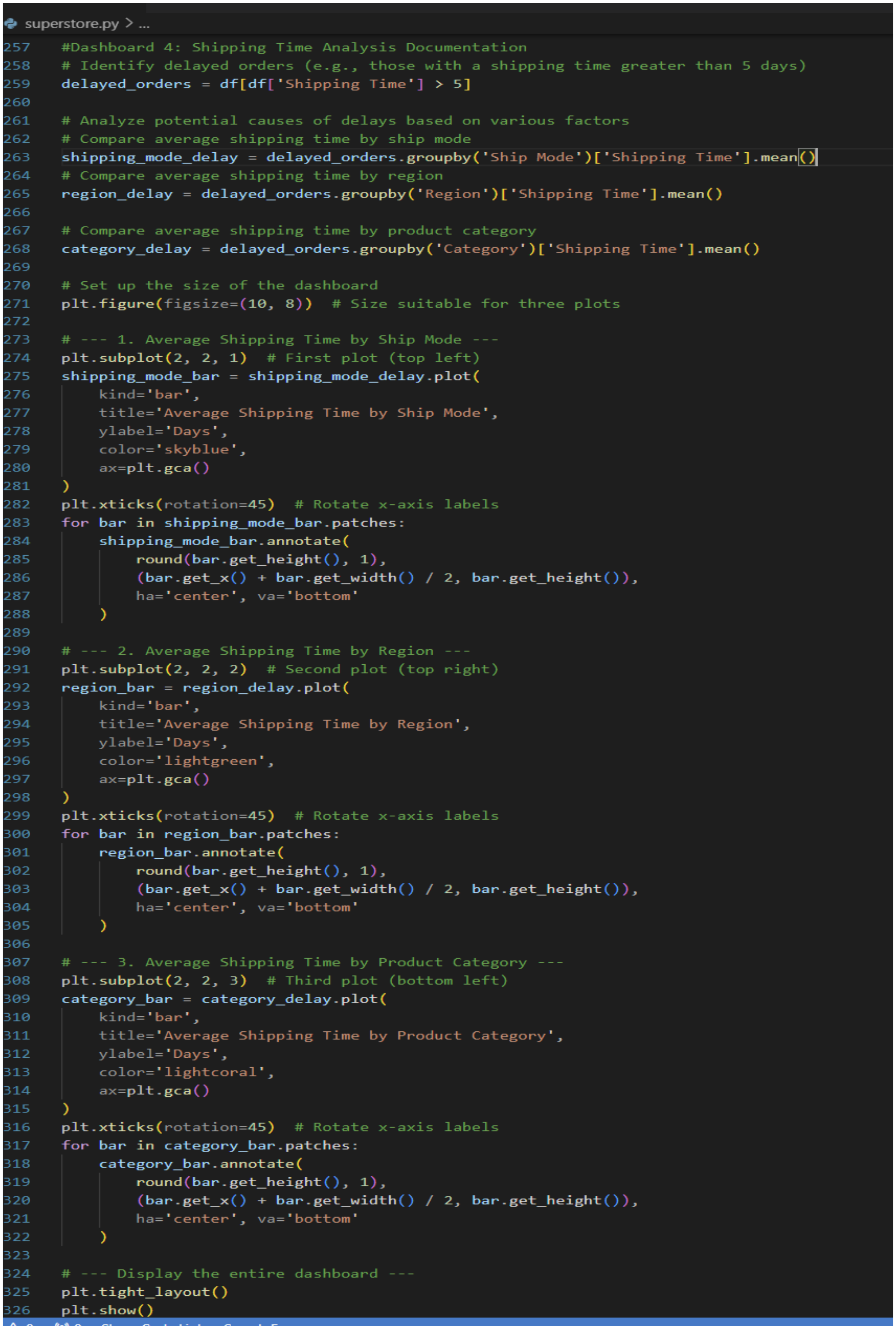
Business Impact

The findings from this analysis can significantly impact the business in the following ways:

- **Improved Customer Satisfaction**: By addressing shipping delays, the company can enhance customer experience and loyalty.

- **Operational Efficiency**: Identifying bottlenecks in the shipping process can lead to optimized logistics and resource allocation.
- **Informed Decision-Making**: The insights gained can assist management in making data-driven decisions regarding shipping partnerships, regional logistics strategies, and inventory management.





```python
#Dashboard 4: Shipping Time Analysis Documentation
# Identify delayed orders (e.g., those with a shipping time greater than 5 days)
delayed_orders = df[df['Shipping Time'] > 5]

# Analyze potential causes of delays based on various factors
# Compare average shipping time by ship mode
shipping_mode_delay = delayed_orders.groupby('Ship Mode')['Shipping Time'].mean()
# Compare average shipping time by region
region_delay = delayed_orders.groupby('Region')['Shipping Time'].mean()

# Compare average shipping time by product category
category_delay = delayed_orders.groupby('Category')['Shipping Time'].mean()

# Set up the size of the dashboard
plt.figure(figsize=(10, 8))  # Size suitable for three plots

# --- 1. Average Shipping Time by Ship Mode ---
plt.subplot(2, 2, 1)  # First plot (top left)
shipping_mode_bar = shipping_mode_delay.plot(
    kind='bar',
    title='Average Shipping Time by Ship Mode',
    ylabel='Days',
    color='skyblue',
    ax=plt.gca()
)
plt.xticks(rotation=45)  # Rotate x-axis labels
for bar in shipping_mode_bar.patches:
    shipping_mode_bar.annotate(
        round(bar.get_height(), 1),
        (bar.get_x() + bar.get_width() / 2, bar.get_height()),
        ha='center', va='bottom'
    )

# --- 2. Average Shipping Time by Region ---
plt.subplot(2, 2, 2)  # Second plot (top right)
region_bar = region_delay.plot(
    kind='bar',
    title='Average Shipping Time by Region',
    ylabel='Days',
    color='lightgreen',
    ax=plt.gca()
)
plt.xticks(rotation=45)  # Rotate x-axis labels
for bar in region_bar.patches:
    region_bar.annotate(
        round(bar.get_height(), 1),
        (bar.get_x() + bar.get_width() / 2, bar.get_height()),
        ha='center', va='bottom'
    )

# --- 3. Average Shipping Time by Product Category ---
plt.subplot(2, 2, 3)  # Third plot (bottom left)
category_bar = category_delay.plot(
    kind='bar',
    title='Average Shipping Time by Product Category',
    ylabel='Days',
    color='lightcoral',
    ax=plt.gca()
)
plt.xticks(rotation=45)  # Rotate x-axis labels
for bar in category_bar.patches:
    category_bar.annotate(
        round(bar.get_height(), 1),
        (bar.get_x() + bar.get_width() / 2, bar.get_height()),
        ha='center', va='bottom'
    )

# --- Display the entire dashboard ---
plt.tight_layout()
plt.show()
```

Dashboard 5

Purpose

The purpose of this analysis is to evaluate sales performance across various categories, regions, and product subcategories. By aggregating sales data, the analysis aims to identify top-performing areas and products, providing insights for strategic decision-making and resource allocation.

Key Metrics

The following key metrics are used to assess sales performance:

- Total Sales by Category: The aggregate sales amount for each product category.

- Total Sales by Region: The total sales amount for each geographical region.

- Total Sales by Sub-Category: The total sales amount for each product subcategory.

- Top Products Sold by Region: A breakdown of sales for the top 10 products across different regions.

Methodology

1. Data Collection: A pandas DataFrame named `df` is prepared, containing relevant sales data.

2. Data Aggregation:

   - Category Sales: Total sales are calculated by grouping data based on `Category`.

   - Region Sales: Total sales are calculated by grouping data based on `Region`.

   - Sub-Category Sales: Total sales are calculated by grouping data based on `Sub-Category`.

   - Product and Region Sales: Total sales are calculated for each product within each region.

3. Filtering Top Products: The top 10 products based on total sales are identified, and the sales data is filtered to include only these products for regional analysis.

4. Visualization: Bar charts are created to visualize the aggregated sales data.

Visualization

The analysis results are presented through a dashboard containing four visualizations:

1. Total Sales by Category

   - A bar chart displaying total sales amounts for each product category.

2. Total Sales by Region

   - A bar chart presenting total sales amounts across various geographical regions.

3. Total Sales by Sub-Category

   - A bar chart illustrating total sales amounts for each product subcategory without annotations.

4. Top 10 Products Sales by Region

   - A stacked bar chart showing the number of products sold for the top 10 products across different regions.

Chart Details

- Each bar chart includes annotations that display the total sales amounts above the bars for clarity.

- The stacked bar chart includes a legend that identifies product names for easy comparison.

Insights

The analysis provides valuable insights, including:

- Top Categories and Regions: Identification of which product categories and regions generate the highest sales, aiding in targeted marketing strategies.

- Product Performance: Understanding which products perform well across different regions, allowing for focused inventory and supply chain management.

- Sub-Category Contributions: Insights into how subcategories contribute to overall sales, guiding product development and promotional efforts.

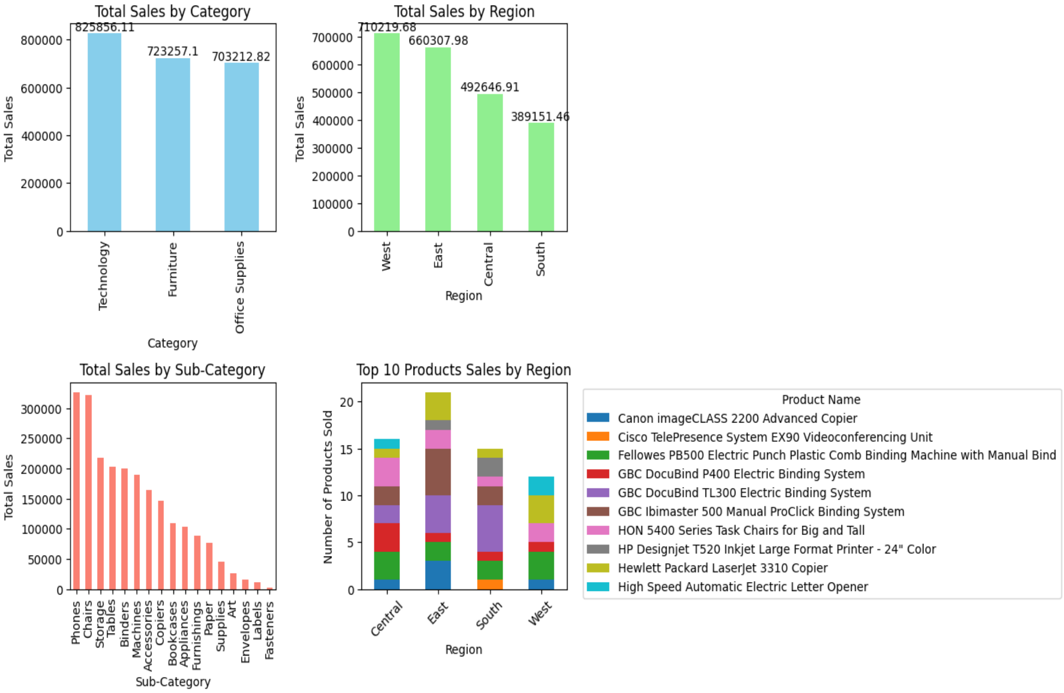
Business Impact

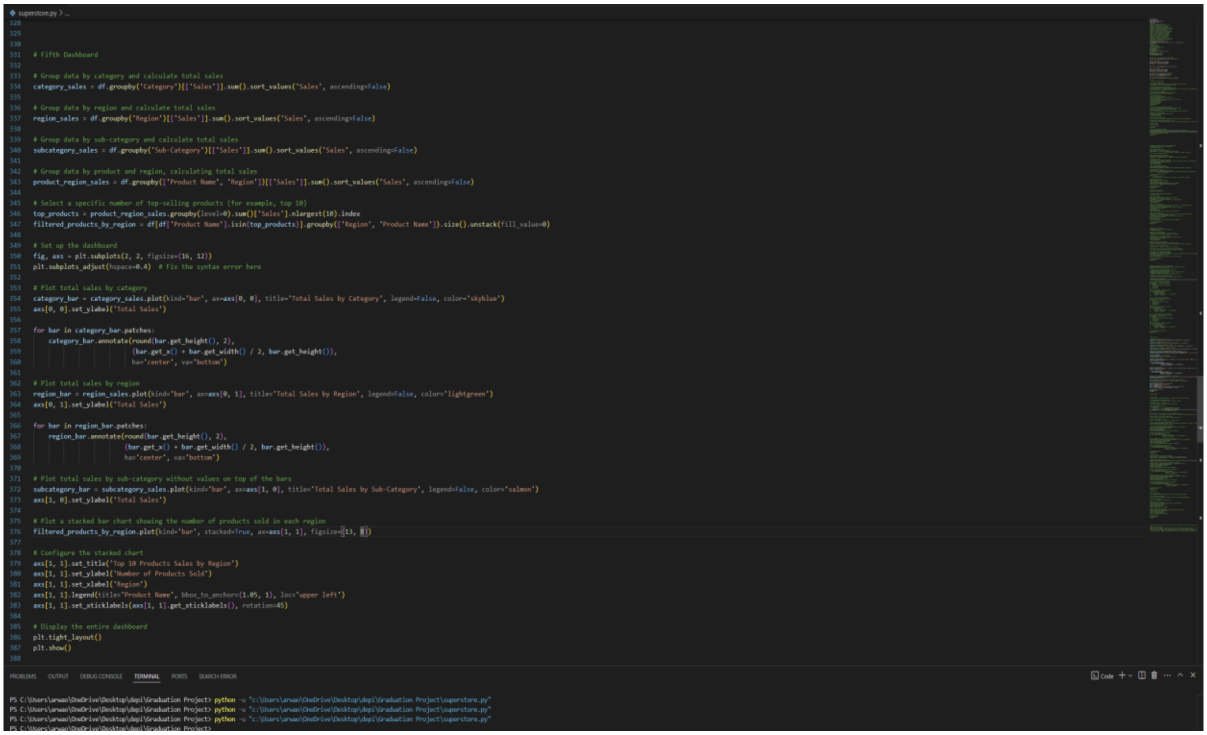
The findings from this analysis can significantly impact the business in the following ways:

- Enhanced Sales Strategies: By understanding sales dynamics, the business can implement more effective sales and marketing strategies.

- Optimized Inventory Management: Insights into top-selling products help in better inventory planning and reduce overstock situations.

- Improved Customer Targeting: The ability to identify top-performing regions allows for tailored marketing campaigns to maximize sales in those areas.





```python
# Fifth Dashboard

# Group data by category and calculate total sales
category_sales = df.groupby('Category')[['Sales']].sum().sort_values('Sales', ascending=False)

# Group data by region and calculate total sales
region_sales = df.groupby('Region')[['Sales']].sum().sort_values('Sales', ascending=False)

# Group data by sub-category and calculate total sales
subcategory_sales = df.groupby('Sub-Category')[['Sales']].sum().sort_values('Sales', ascending=False)

# Group data by product and region, calculating total sales
product_region_sales = df.groupby(['Product Name', 'Region'])[['Sales']].sum().sort_values('Sales', ascending=False)

# Select a specific number of top-selling products (for example, top 10)
top_products = product_region_sales.groupby(level=0).sum()['Sales'].nlargest(10).index
filtered_products_by_region = df[df['Product Name'].isin(top_products)].groupby(['Region', 'Product Name']).size().unstack(fill_value=0)

# Set up the dashboard
fig, axs = plt.subplots(2, 2, figsize=(16, 12))
plt.subplots_adjust(hspace=0.4)  # Fix the syntax error here

# Plot total sales by category
category_bar = category_sales.plot(kind='bar', ax=axs[0, 0], title='Total Sales by Category', legend=False, color='skyblue')
axs[0, 0].set_ylabel('Total Sales')

for bar in category_bar.patches:
    category_bar.annotate(round(bar.get_height(), 2),
                          (bar.get_x() + bar.get_width() / 2, bar.get_height()),
                          ha='center', va='bottom')

# Plot total sales by region
region_bar = region_sales.plot(kind='bar', ax=axs[0, 1], title='Total Sales by Region', legend=False, color='lightgreen')
axs[0, 1].set_ylabel('Total Sales')

for bar in region_bar.patches:
    region_bar.annotate(round(bar.get_height(), 2),
                        (bar.get_x() + bar.get_width() / 2, bar.get_height()),
                        ha='center', va='bottom')

# Plot total sales by sub-category without values on top of the bars
subcategory_bar = subcategory_sales.plot(kind='bar', ax=axs[1, 0], title='Total Sales by Sub-Category', legend=False, color='salmon')
axs[1, 0].set_ylabel('Total Sales')

# Plot a stacked bar chart showing the number of products sold in each region
filtered_products_by_region.plot(kind='bar', stacked=True, ax=axs[1, 1], figsize=(13, 8))

# Configure the stacked chart
axs[1, 1].set_title('Top 10 Products Sales by Region')
axs[1, 1].set_ylabel('Number of Products Sold')
axs[1, 1].set_xlabel('Region')
axs[1, 1].legend(title='Product Name', bbox_to_anchor=(1.05, 1), loc='upper left')
axs[1, 1].set_xticklabels(axs[1, 1].get_xticklabels(), rotation=45)

# Display the entire dashboard
plt.tight_layout()
plt.show()
```

```
PS C:\Users\arwao\OneDrive\Desktop\depi\Graduation Project> python -u "c:\Users\arwao\OneDrive\Desktop\depi\Graduation Project\superstore.py"
PS C:\Users\arwao\OneDrive\Desktop\depi\Graduation Project> python -u "c:\Users\arwao\OneDrive\Desktop\depi\Graduation Project\superstore.py"
PS C:\Users\arwao\OneDrive\Desktop\depi\Graduation Project> python -u "c:\Users\arwao\OneDrive\Desktop\depi\Graduation Project\superstore.py"
PS C:\Users\arwao\OneDrive\Desktop\depi\Graduation Project>
```

Dashboard: 6

Purpose

The main objective of this dashboard is to analyze customer behavior and operational efficiency by evaluating shipping preferences, sales performance by shipping mode, and delays in the shipping process. Understanding these metrics helps businesses improve customer satisfaction, optimize shipping methods, and reduce delayed deliveries, ultimately enhancing profitability and service quality.

Key Metrics

1. **Customer Shipping Mode Preferences**: The proportion of customers who use each shipping mode (Standard Class, First Class, Second Class, and Same Day).

2. **Average Sales by Shipping Mode**: The average revenue generated for each shipping mode to assess the financial impact of different shipping options.

3. **Delayed Orders by Shipping Mode**: The number of delayed orders for each shipping mode to identify areas of inefficiency and possible improvements.

Methodology

1. **Data Preparation**:

    ○ The dataset is grouped by **Customer ID** and **Ship Mode** to understand customer preferences and behavior.

    ○ The data is also grouped by **Ship Mode** to calculate unique customer counts, average sales, and the number of delayed orders.

2. **Shipping Delays**:

    ○ A business-defined threshold for maximum shipping time is set for each shipping mode (e.g., 5 days for Standard Class, 1 day for Same Day).

    ○ A new column Is Delayed is created to indicate if the actual shipping time exceeded the maximum for that shipping mode.

3. **Visualization**:

- A **pie chart** is used to visualize the proportion of customers using each shipping mode, highlighting preferences.
- A **bar chart** displays the **average sales** by shipping mode, showing how different shipping methods affect revenue.
- Another **bar chart** illustrates the number of delayed orders by shipping mode to identify inefficiencies and high-risk shipping methods.

Visualizations

1. **Pie Chart**:
    - Title: *Customer Shipping Mode Preferences*
    - Shows the proportion of customers who prefer each shipping mode.
2. **Bar Chart 1**:
    - Title: *Average Sales by Shipping Mode*
    - X-axis: Shipping Mode (Standard Class, First Class, etc.)
    - Y-axis: Average Sales in dollars
    - Annotates the exact sales value on top of each bar for clarity.
3. **Bar Chart 2**:
    - Title: *Delayed Orders by Shipping Mode*
    - X-axis: Shipping Mode
    - Y-axis: Number of Delayed Orders
    - Annotates the number of delayed orders on top of each bar to highlight delays.

Insights

1. **Customer Preferences**:
    - The pie chart helps in identifying the most and least popular shipping modes among customers. For instance, if most customers prefer *Standard Class*, this might indicate that the business could focus more resources on optimizing this option.

2. **Sales Performance**:

   - The average sales per shipping mode bar chart reveals which shipping modes generate the highest revenue. For example, if *Same Day* shipping leads to higher average sales, it might be beneficial to promote this option more heavily, despite potentially higher operational costs.

3. **Operational Delays**:

   - The delayed orders bar chart helps identify shipping modes prone to delays. If *Standard Class* has the highest number of delays, this may indicate issues with logistical coordination, warehouse management, or external courier services. This insight could prompt operational changes to improve efficiency and reduce delivery times.

Business Impact

1. **Customer Satisfaction**:

   - By analyzing customer shipping preferences, the company can align its offerings with customer needs, which can lead to higher satisfaction rates, loyalty, and repeat business.

2. **Revenue Optimization**:

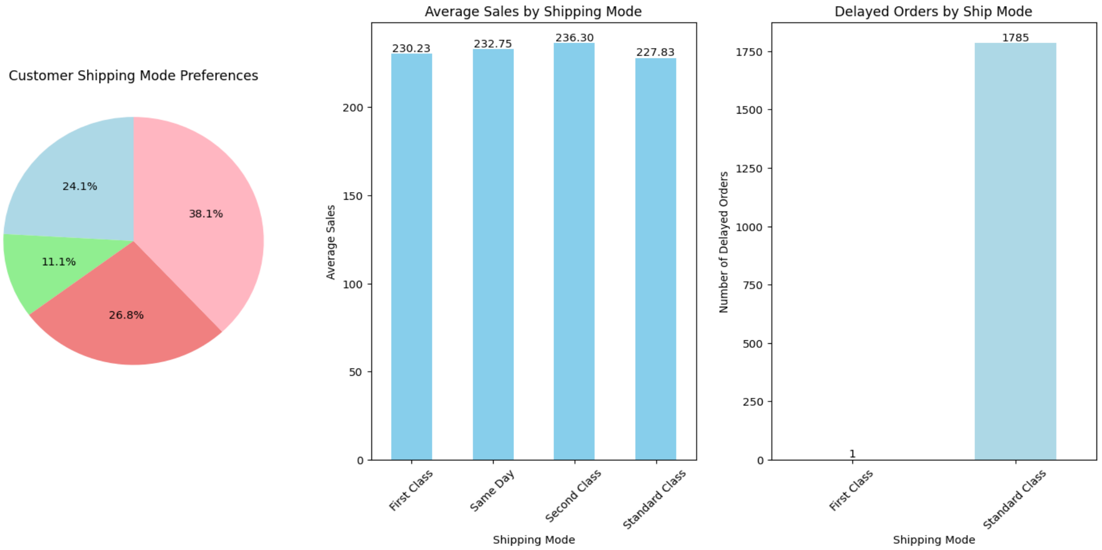
   - The average sales analysis helps in understanding which shipping methods bring in the most revenue. This insight allows businesses to tailor promotions and suggest shipping options that might drive higher sales while balancing costs.

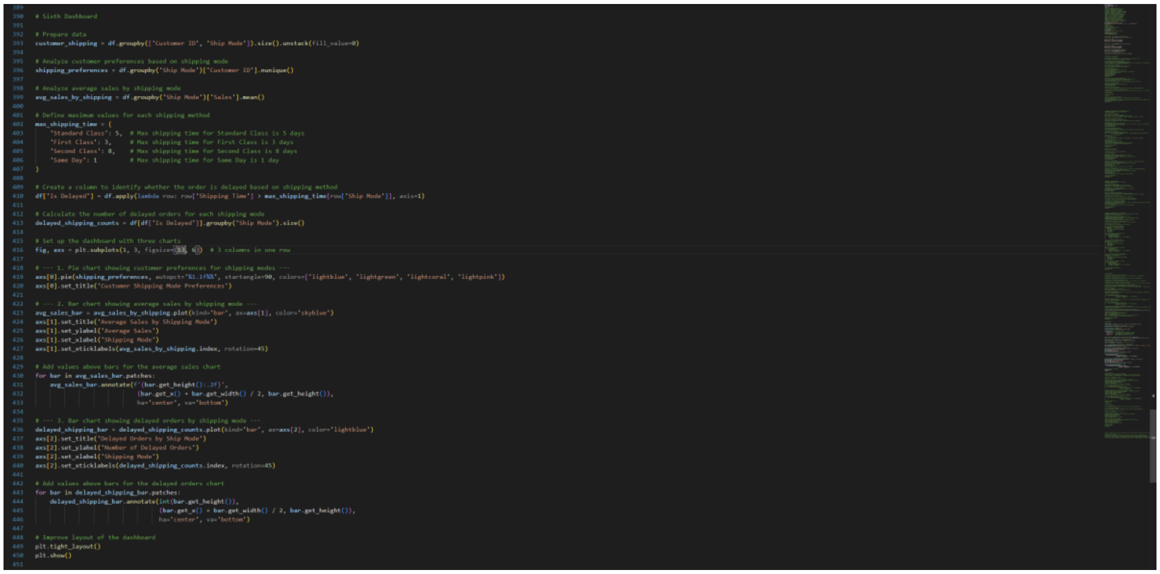
3. **Reduced Delays and Operational Efficiency**:

   - Identifying the shipping modes with the most delays enables the company to focus on improving its logistics for those modes. Reducing delays can lead to fewer customer complaints, reduced costs of dealing with issues, and improved customer retention.

4. **Strategic Decision-Making**:

   - The business can use this data to make informed decisions on pricing, marketing, and logistical operations. For example, if a particular shipping mode is both preferred and highly profitable, efforts to streamline and scale up this service could provide a competitive advantage.





```
# Sixth Dashboard

# Prepare data
customer_shipping = df.groupby(['Customer ID', 'Ship Mode']).size().unstack(fill_value=0)

# Analyze customer preferences based on shipping mode
shipping_preferences = df.groupby('Ship Mode')['Customer ID'].nunique()

# Analyze average sales by shipping mode
avg_sales_by_shipping = df.groupby('Ship Mode')['Sales'].mean()

# Define maximum values for each shipping method
max_shipping_time = {
    'Standard Class': 5,  # Max shipping time for Standard Class is 5 days
    'First Class': 3,     # Max shipping time for First Class is 3 days
    'Second Class': 8,    # Max shipping time for Second Class is 8 days
    'Same Day': 1         # Max shipping time for Same Day is 1 day
}

# Create a column to identify whether the order is delayed based on shipping method
df['Is Delayed'] = df.apply(lambda row: row['Shipping Time'] > max_shipping_time[row['Ship Mode']], axis=1)

# Calculate the number of delayed orders for each shipping mode
delayed_shipping_counts = df[df['Is Delayed']].groupby('Ship Mode').size()

# Set up the dashboard with three charts
fig, axs = plt.subplots(1, 3, figsize=(13, 6))  # 3 columns in one row

# --- 1. Pie chart showing customer preferences for shipping modes ---
axs[0].pie(shipping_preferences, autopct='%1.1f%%', startangle=90, colors=['lightblue', 'lightgreen', 'lightcoral', 'lightpink'])
axs[0].set_title('Customer Shipping Mode Preferences')

# --- 2. Bar chart showing average sales by shipping mode ---
avg_sales_bar = avg_sales_by_shipping.plot(kind='bar', ax=axs[1], color='skyblue')
axs[1].set_title('Average Sales by Shipping Mode')
axs[1].set_ylabel('Average Sales')
axs[1].set_xlabel('Shipping Mode')
axs[1].set_xticklabels(avg_sales_by_shipping.index, rotation=45)

# Add values above bars for the average sales chart
for bar in avg_sales_bar.patches:
    avg_sales_bar.annotate(f'{bar.get_height():.2f}',
                           (bar.get_x() + bar.get_width() / 2, bar.get_height()),
                           ha='center', va='bottom')

# --- 3. Bar chart showing delayed orders by shipping mode ---
delayed_shipping_bar = delayed_shipping_counts.plot(kind='bar', ax=axs[2], color='lightblue')
axs[2].set_title('Delayed Orders by Ship Mode')
axs[2].set_ylabel('Number of Delayed Orders')
axs[2].set_xlabel('Shipping Mode')
axs[2].set_xticklabels(delayed_shipping_counts.index, rotation=45)

# Add values above bars for the delayed orders chart
for bar in delayed_shipping_bar.patches:
    delayed_shipping_bar.annotate(int(bar.get_height()),
                                  (bar.get_x() + bar.get_width() / 2, bar.get_height()),
                                  ha='center', va='bottom')

# Improve layout of the dashboard
plt.tight_layout()
plt.show()
```

Dashboard 7

Purpose

This dashboard is designed to analyze the usage of different shipping modes across various regions and product categories. It helps in identifying which shipping modes are most frequently used in specific regions or categories, providing actionable insights for logistics and operational strategies.

Key Metrics

1. **Shipping Mode Usage by Region**: The number of times each shipping mode (Standard Class, First Class, Second Class, Same Day) is used in each region.
2. **Shipping Mode Usage by Category**: The number of times each shipping mode is used across product categories.

Methodology

1. **Data Preparation**:
   - **Shipping Mode by Region**: The dataset is grouped by both **Region** and **Ship Mode** to aggregate the number of shipments per shipping mode for each region.
   - **Shipping Mode by Category**: Similarly, the dataset is grouped by **Category** and **Ship Mode** to find the number of shipments per mode for each product category.
2. **Visualization**:
   - **Stacked Bar Charts** are used for visualizing both shipping mode usage by region and category. This allows us to compare the contribution of each shipping mode within a region or category and understand usage patterns.

Visualizations

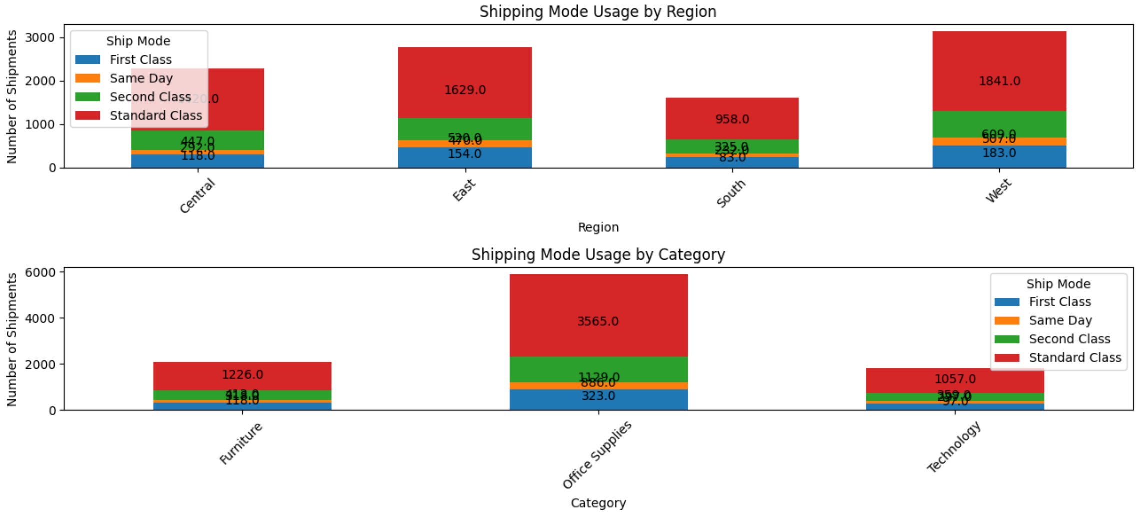
1. **Bar Chart 1**:
   - Title: *Shipping Mode Usage by Region*
   - X-axis: Regions (e.g., East, West, Central, South)
   - Y-axis: Number of Shipments
   - The chart is **stacked** by shipping mode (Standard Class, First Class, etc.), showing how frequently each mode is used in different regions.
   - **Annotations**: The exact number of shipments is added on top of each bar for clarity.
2. **Bar Chart 2**:
   - Title: *Shipping Mode Usage by Category*
   - X-axis: Product Categories (e.g., Furniture, Office Supplies, Technology)
   - Y-axis: Number of Shipments
   - The chart is **stacked** by shipping mode, showing which modes are popular in each category.
   - **Annotations**: The number of shipments is also added on top of each bar to provide a clear understanding of shipping frequency by category.

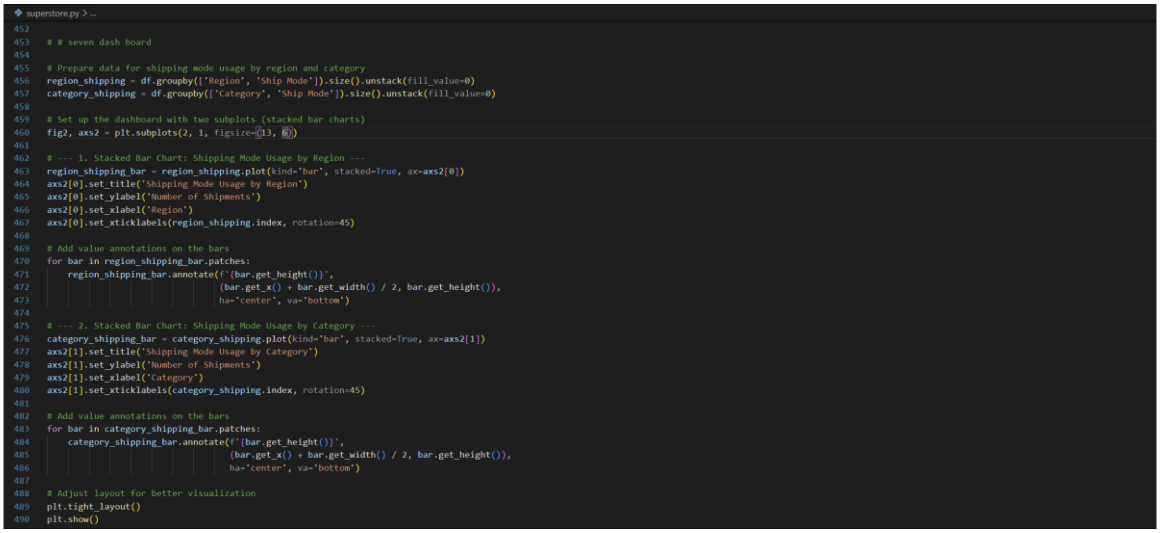
Insights

1. **Regional Shipping Preferences**:
   - The stacked bar chart showing **Shipping Mode Usage by Region** highlights which shipping modes are most commonly used in specific regions. For example, if *Standard Class* is most prevalent in the West region, it may indicate cost-sensitive customers or logistical constraints that favor slower shipping options.
2. **Category-Based Shipping Patterns**:
   - The second stacked bar chart reveals which shipping modes are popular in different product categories. For instance, if *Same Day* shipping is frequently used for technology products, it suggests that customers in this category may prioritize fast deliveries, possibly due to the urgency associated with electronics.

Business Impact

1. **Operational Efficiency**:
   - Understanding regional shipping patterns helps the business optimize warehouse locations, route planning, and carrier selection based on the most common shipping modes used in each region.
2. **Category-Specific Shipping Insights**:
   - By analyzing how different shipping modes are used for various product categories, the business can tailor its shipping options. For instance, categories with higher usage of fast shipping options like *Same Day* can be prioritized for rapid fulfillment and targeted promotions.
3. **Cost Optimization**:
   - This dashboard allows decision-makers to see if certain shipping modes are overused in regions or categories where cheaper alternatives could suffice. By adjusting shipping strategies, the company can save costs while maintaining customer satisfaction.





```python

# # seven dash board

# Prepare data for shipping mode usage by region and category
region_shipping = df.groupby(['Region', 'Ship Mode']).size().unstack(fill_value=0)
category_shipping = df.groupby(['Category', 'Ship Mode']).size().unstack(fill_value=0)

# Set up the dashboard with two subplots (stacked bar charts)
fig2, axs2 = plt.subplots(2, 1, figsize=(13, 6))

# --- 1. Stacked Bar Chart: Shipping Mode Usage by Region ---
region_shipping_bar = region_shipping.plot(kind='bar', stacked=True, ax=axs2[0])
axs2[0].set_title('Shipping Mode Usage by Region')
axs2[0].set_ylabel('Number of Shipments')
axs2[0].set_xlabel('Region')
axs2[0].set_xticklabels(region_shipping.index, rotation=45)

# Add value annotations on the bars
for bar in region_shipping_bar.patches:
    region_shipping_bar.annotate(f'{bar.get_height()}',
                                 (bar.get_x() + bar.get_width() / 2, bar.get_height()),
                                 ha='center', va='bottom')

# --- 2. Stacked Bar Chart: Shipping Mode Usage by Category ---
category_shipping_bar = category_shipping.plot(kind='bar', stacked=True, ax=axs2[1])
axs2[1].set_title('Shipping Mode Usage by Category')
axs2[1].set_ylabel('Number of Shipments')
axs2[1].set_xlabel('Category')
axs2[1].set_xticklabels(category_shipping.index, rotation=45)

# Add value annotations on the bars
for bar in category_shipping_bar.patches:
    category_shipping_bar.annotate(f'{bar.get_height()}',
                                   (bar.get_x() + bar.get_width() / 2, bar.get_height()),
                                   ha='center', va='bottom')

# Adjust layout for better visualization
plt.tight_layout()
plt.show()
```

Dashboard 8: Sales by Category and Sub-Category

Purpose:

The primary objective of this dashboard is to visualize the total sales distribution across different product **categories** and **sub-categories**. This helps identify which categories and sub-categories contribute the most to overall sales and highlights opportunities for further analysis or improvement.

Key Metrics:

1. **Total Sales by Category:** Displays total sales for each high-level product category.
2. **Total Sales by Sub-Category:** Shows total sales for each sub-category, providing a more granular view of sales performance.

Methodology:

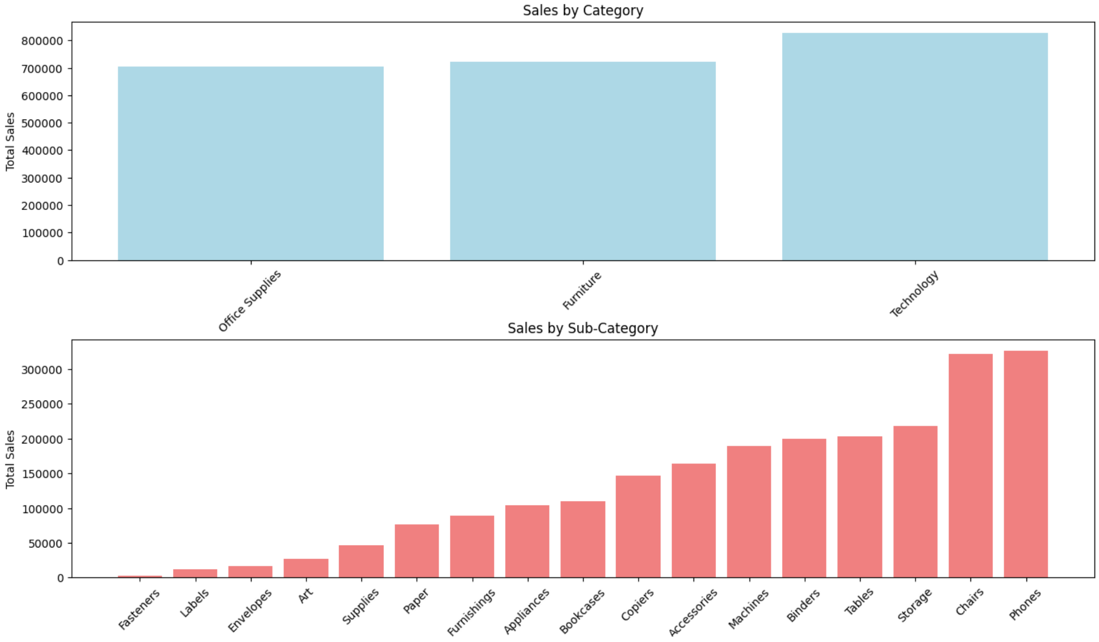
1. **Data Grouping and Aggregation:**
   - The data is grouped by Category and Sub-Category.
   - The total sales for each group are calculated using the sum() function.
   - The values are sorted in ascending order to create a better visual representation.
2. **Visualization:**
   - Two bar charts are used to represent the total sales by category and sub-category.
   - Each bar's height represents the total sales for the corresponding category or sub-category.
   - **Bar Chart 1:** Shows the sales distribution across high-level categories, providing a broader perspective.
   - **Bar Chart 2:** Provides a detailed view of sales at the sub-category level, allowing for more specific insights.

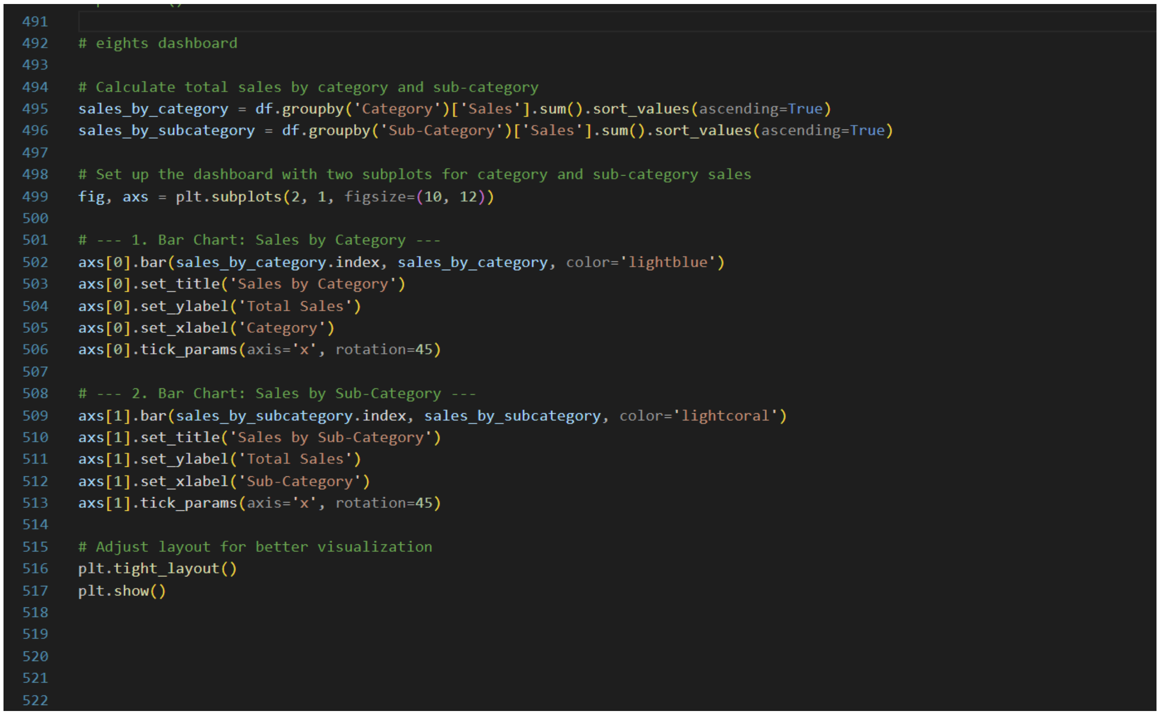
Insights:

- The **category sales chart** provides a clear view of how overall sales are distributed across broad product categories.
- The **sub-category sales chart** gives a more detailed breakdown, which can highlight the specific products driving revenue within each category.

Business Impact:

- This dashboard helps business leaders and decision-makers identify which product lines are the most profitable, allowing them to make informed decisions about inventory management, marketing focus, and sales strategies.
- It also highlights underperforming categories or sub-categories, offering opportunities to adjust marketing efforts or product offerings to boost sales.





```python

# eights dashboard

# Calculate total sales by category and sub-category
sales_by_category = df.groupby('Category')['Sales'].sum().sort_values(ascending=True)
sales_by_subcategory = df.groupby('Sub-Category')['Sales'].sum().sort_values(ascending=True)

# Set up the dashboard with two subplots for category and sub-category sales
fig, axs = plt.subplots(2, 1, figsize=(10, 12))

# --- 1. Bar Chart: Sales by Category ---
axs[0].bar(sales_by_category.index, sales_by_category, color='lightblue')
axs[0].set_title('Sales by Category')
axs[0].set_ylabel('Total Sales')
axs[0].set_xlabel('Category')
axs[0].tick_params(axis='x', rotation=45)

# --- 2. Bar Chart: Sales by Sub-Category ---
axs[1].bar(sales_by_subcategory.index, sales_by_subcategory, color='lightcoral')
axs[1].set_title('Sales by Sub-Category')
axs[1].set_ylabel('Total Sales')
axs[1].set_xlabel('Sub-Category')
axs[1].tick_params(axis='x', rotation=45)

# Adjust layout for better visualization
plt.tight_layout()
plt.show()
```