### **Superstore Sales Data Analysis: Comprehensive Project Report**

#### **Team Members:**

* Nourhan Kazem
* Mohamed Abdel Menaeim
* Virginia Tawfik

### **1. Introduction**

The objective of this project was to analyze the Superstore sales data, focusing on sales trends, customer behavior, and product performance across various dimensions like product categories, regions, and shipping modes. By employing **Python** for data preprocessing, statistical analysis, and predictive modeling, and **Tableau** for rich data visualizations, the goal was to offer strategic insights that would guide future business decisions.

This project is divided into three major phases:

1. **Data Preprocessing**: Cleaning and preparing the dataset for analysis.
2. **Data Analysis and Visualization**: Using Python to analyze and visualize the data.
3. **Forecasting and Future Trends**: Building a forecasting model to predict future sales using machine learning techniques.

### **2. Tools and Technologies Used**

* **Python**: Used for data cleaning, analysis, and machine learning.
* **Tableau**: Used for creating interactive dashboards that visualize the trends and patterns.
* **Python Libraries**:
  + **Pandas** for data manipulation and cleaning.
  + **Matplotlib** and **Seaborn** for visualizations.
  + **Scikit-learn** for building predictive models.
  + **Folium** for geospatial visualizations.

### **3. Project Workflow: Comprehensive Breakdown**

The project workflow involved multiple phases, from initial data preprocessing to advanced analysis, visualizations, and forecasting. Each phase was meticulously designed to extract meaningful insights from the Superstore dataset and transform these insights into actionable business strategies. The workflow was designed to ensure a step-by-step, systematic approach, where each task builds on the previous one to achieve a well-rounded analysis.

#### **3.1 Data Preprocessing**

**Objective**:  
Before performing any analysis, we needed to ensure the data was clean, structured, and free from inconsistencies. The goal was to prepare a dataset that could be reliably used for exploratory data analysis (EDA), statistical analysis, and machine learning.

**Steps Involved**:

1. **Data Loading**:
   * We began by loading the dataset using **Pandas**, a Python library that makes handling large datasets efficient.
   * The dataset was stored in a .csv file, and it contained columns such as Order Date, Sales, Profit, Category, Customer Segment, Region, and Ship Mode.

**Code**:

import pandas as pd df = pd.read\_csv('superstore\_data.csv')

**PowerPoint Figures**:

* The first slide (Slide 2) in the presentation highlights this as part of the data preprocessing process.

2. **Handling Missing Values**:

* We inspected the dataset for missing values in key columns such as Sales, Profit, and Order Date. Missing values, if left unchecked, can lead to inaccurate analysis or biased results.
* After identifying the missing values, we dropped the rows that contained missing data in essential columns, since the dataset had a manageable amount of missing entries.

**Code**:

df.dropna(inplace=True)

**PowerPoint Figures**:

* On Slide 3, the presentation emphasizes that all missing values were handled properly.

3. **Checking for Duplicates**:

* Duplicates in the dataset were checked using the Pandas function .duplicated(). Duplicates could artificially inflate sales, profit, or any other key metric if not removed.
* No duplicate rows were found, which indicated that the dataset was free from redundancy.

**Code**:

duplicates = df.duplicated().sum() if duplicates == 0: print("No duplicates found.")

**PowerPoint Figures**:

* Slide 4 explicitly mentions that no duplicate rows were found during the data cleaning phase.

4. **Outlier Detection and Removal**:

* Outlier detection was performed to identify any extreme data points that could skew the results. For instance, very high sales values in specific categories or regions could bias the analysis.
* We used **boxplots** to detect outliers in Sales and Profit data. The 5-number summary statistics (min, Q1, median, Q3, max) were also calculated to aid in the outlier analysis.
* Surprisingly, no significant outliers were found in the Sales and Profit data. This indicates that the dataset is well-behaved and representative of the business's actual performance.

**Code**:

import seaborn as sns sns.boxplot(x=df['Sales']) sns.boxplot(x=df['Profit'])

**PowerPoint Figures**:

* Slides 6 and 7 display the boxplots for Sales and Profit, showing that the data was clean and free from outliers. The boxplots visually demonstrated that the dataset had no extreme values that needed removal.

5. **Feature Selection (Correlation Matrix)**:

* To focus our analysis on the most relevant information, we performed feature selection. We kept columns such as Sales, Profit, Quantity, Category, Region, and Ship Mode, as these were critical to understanding business performance.
* Less relevant columns, such as Postal Code, were excluded from the analysis as they did not contribute directly to the objectives of the study.

**Code**:

df = df[['Order Date', 'Sales', 'Profit', 'Quantity', 'Discount', 'Category', 'Region']]

**PowerPoint Figures**:

* Slide 8 describes how non-numeric columns and less important variables were excluded from the analysis, focusing on the primary metrics.
* **Relation between correlation and summary statistics:**

**1. Correlation Matrix:**

Purpose: A correlation matrix helps you understand the relationships between different numeric variables. It tells you how much one variable is related to another (with values between -1 and 1), where:

+1 means a perfect positive correlation,

-1 means a perfect negative correlation, and

0 means no correlation.

Why do it first? It gives an immediate understanding of the data’s internal structure. By seeing which variables are strongly correlated, you can gain insight into how features influence each other. For example, high correlation between two features may indicate multicollinearity, which can be important when developing predictive models.

**2. Summary Statistics:**

Purpose: Summary statistics provide descriptive statistics like mean, standard deviation, minimum, maximum, and quartiles for each numeric variable. This tells you the central tendencies and variability of each feature independently.

Why after correlation? After understanding the relationships between features from the correlation matrix, the summary statistics help you get more specific insights into each variable (such as its range, average values, or spread). You can also focus more on the features that showed stronger correlations, guiding further analysis or model development.

Key Insights From This Order:

Correlation Matrix First: Helps identify relationships between variables (e.g., sales and profit), guiding you to areas of potential interest or concern.

Summary Statistics Next: Provides a detailed overview of individual variables, helping you understand how spread out or centralized the data is, and informing decisions such as normalization or transformations.

In conclusion, the correlation matrix gives you a high-level understanding of variable relationships, while summary statistics give a detailed look at the data itself. Doing them in this order is logical if you want to explore relationships before digging into each feature’s detailed properties.

* **Insights from the Correlation Matrix:**

The correlation matrix reveals the strength and direction of the relationships between the variables Sales, Quantity, Discount, and Profit. Let’s break it down:

**Sales and Profit:**

Correlation: 0.479 (moderate positive correlation).

Insight: As sales increase, profit tends to increase as well, but not perfectly. There are other factors influencing profit, like cost, discounting, or product mix.

Sales and Quantity:

Correlation: 0.200 (weak positive correlation).

Insight: While higher quantities might lead to higher sales, the relationship is not strong. This could suggest that selling large quantities doesn't always mean higher sales values, possibly due to discounts or lower-value items being sold in bulk.

Sales and Discount:

Correlation: -0.028 (very weak negative correlation).

Insight: The relationship between sales and discounts is nearly non-existent, indicating that offering discounts does not directly drive higher sales in this dataset.

Quantity and Profit:

Correlation: 0.066 (very weak positive correlation).

Insight: Selling more items doesn't strongly correlate with higher profits, suggesting that quantity alone isn't a key driver of profitability, perhaps due to pricing strategies or product types.

**Discount and Profit:**

Correlation: -0.219 (weak negative correlation).

Insight: There is a weak inverse relationship between discounts and profit. As discounts increase, profit tends to decrease slightly, which makes sense since higher discounts can reduce margins.

Discount and Quantity:

Correlation: 0.009 (almost no correlation).

Insight: Discounts don’t seem to significantly influence the quantity sold, meaning offering discounts doesn’t necessarily increase sales volume in this dataset.

Insights from the Summary Statistics:

Sales:

Mean Sales: $229.86, but the standard deviation is high ($623.24), indicating a wide range of sales values.

Min Sales: $0.44, and Max Sales: $22,638.48, showing a huge spread. The median (50% quartile) is $54.49, meaning half the sales are below this value.

Insight: The distribution of sales is highly skewed, with most sales being much lower than the maximum.

**Quantity:**

Mean Quantity: 3.79 units per transaction with a standard deviation of 2.23.

Insight: The quantity sold per order is relatively low, and most orders involve small quantities.

Discount:

Mean Discount: 0.156 (or 15.6%), with a max discount of 80%.

Insight: Discounts are applied moderately, with a median of 20%, but not all transactions have discounts (as seen in the 25% and 50% quartiles where the discount is 0%).

Profit:

Mean Profit: $28.66, with a high standard deviation ($234.26), meaning profit values vary greatly. The range is also large, with a minimum profit of -$6,599.98 (loss) and a maximum of $8,399.98.

Insight: There are significant profit fluctuations, with many orders likely generating small profits or even losses (as suggested by the negative minimum value).

**Key Takeaways:**

1- Moderate Positive Correlation Between Sales and Profit: Higher sales generally lead to higher profits, but this is not always consistent.

2- Weak Relationship Between Quantity and Profit: Selling more items doesn't guarantee higher profits, likely due to the effect of pricing, product mix, or discounts.

3- Negative Impact of Discounts on Profit: Higher discounts tend to slightly reduce profits, indicating that aggressive discounting may hurt profitability.

4- High Variability in Sales and Profit: Both metrics have a wide range, indicating that some transactions are very large or profitable, but most are relatively small.

5- Discounts Don't Drive Quantity: Offering discounts doesn’t seem to significantly affect the quantity sold.

6- This analysis can help guide business decisions, such as reassessing discounting strategies or focusing on increasing sales of higher-margin products.

#### **3.2 Descriptive Analysis**

**Objective**:  
The goal of this phase was to extract key business insights by analyzing the dataset’s central trends. We aimed to answer questions about performance across different regions, product categories, shipping modes, and customer segments.

**Steps Involved**:

1. **Sales Performance by Product Category**:
   * The dataset was grouped by the Category column to calculate total sales for each product category: **Furniture**, **Office Supplies**, and **Technology**.
   * A **bar chart** was created to visualize sales performance by category, helping us understand which products generated the most revenue.

**Code**:

df.groupby('Category')['Sales'].sum().plot(kind='bar', title='Total Sales by Product Category')

**PowerPoint Figures**:

* Slide 9 presents the bar chart for sales by product category. It shows that **Office Supplies** had the largest volume of sales, while **Technology** had the highest profit margins.

2. **Profit and Sales by Region**:

* We calculated total Sales and Profit for each region (West, East, Central, South) and visualized the results using **pie charts**. This gave us a clear view of how each region contributed to overall business performance.
* This was a critical analysis as it allowed us to understand which regions were most profitable and which might need further attention or investment.

**Code**:

df.groupby('Region')[['Sales', 'Profit']].sum().plot.pie(subplots=True, title='Sales and Profit by Region')

**PowerPoint Figures**:

* Slide 10 shows the pie charts for both sales and profit by region. The **West** region had the highest sales and profit contributions, while the **South** region had the lowest.

3. **Shipping Mode Analysis**:

* To assess the impact of shipping methods on sales, we grouped the data by Ship Mode and calculated total sales for each mode: **First Class**, **Second Class**, **Standard Class**, and **Same Day**.
* A **bar chart** was used to compare the total sales generated by each shipping mode.

**Code**:

df.groupby('Ship Mode')['Sales'].sum().plot(kind='bar', title='Sales by Shipping Mode')

**PowerPoint Figures**:

* + Slide 11 displays the bar chart for sales by shipping mode. It shows that **Standard Class** accounted for the majority of sales, but **Same Day** shipping, though less common, had the potential for higher profit margins.

#### **3.3 Python Charts and Insights**

In this phase, we leveraged Python’s visualization libraries to create detailed charts that offered deeper insights into the dataset. These visualizations were crucial for answering business questions and supporting data-driven decision-making.

### **#1: Sales Performance Across Different Product Categories**

**Objective**:  
To analyze how different product categories perform in terms of total sales. This helps identify which categories are driving the most revenue and where the business should focus its efforts to maximize sales.

**Code Breakdown**:

**1- Grouping Data by Product Category**:

* + The dataset is first grouped by the Category column to aggregate the total sales for each product category. This step helps us understand how each category contributes to overall sales. The sort\_values() function is used to arrange the sales values in ascending order, making it easier to visualize category performance.

**Code**:

category\_sales = dataset\_no\_outliers.groupby('Category')['Sales'].sum().sort\_values()

**Explanation**:

* By grouping the dataset by Category and summing the Sales values, we get the total sales amount for each product category. This allows us to see which product categories (e.g., Technology, Furniture, Office Supplies) are performing better than others in terms of sales.

**2- Creating the Bar Plot**:

* A **bar plot** is then created to visualize the total sales for each product category. The plt.figure(figsize=(10, 6)) ensures the plot is large enough for clarity, while the category\_sales.plot(kind='bar', color='skyblue') line generates the bar plot with a light blue color to make it visually appealing and easy to differentiate between categories.

**Code**:

plt.figure(figsize=(10, 6))

category\_sales.plot(kind='bar', color='skyblue')

plt.title('Sales Performance Across Different Product Categories')

plt.xlabel('Product Category')

plt.ylabel('Total Sales')

plt.xticks(rotation=45)

plt.grid(axis='y', linestyle='--', alpha=0.6)

plt.show()

**Explanation**:

* **Plot Dimensions**: The figsize=(10, 6) creates a figure that is 10 inches wide and 6 inches tall, ensuring the plot is neither too small nor too large for presentation.
* **Bar Plot**: The bar plot visually compares total sales across categories. Each bar represents a product category, and its height corresponds to the total sales generated by that category.
* **Axis Labels and Title**: The plt.title() gives the plot a descriptive title, and plt.xlabel() and plt.ylabel() provide labels for the x- and y-axes, ensuring the chart is easy to interpret.
* **Rotation of Labels**: The plt.xticks(rotation=45) rotates the x-axis labels by 45 degrees, preventing overlap and improving readability.
* **Gridlines**: A grid is added to the y-axis using plt.grid(axis='y', linestyle='--', alpha=0.6), which helps the viewer compare the heights of the bars more easily.

**Visualization and Insights**:

* The final plot visually depicts how each product category performs in terms of total sales. This makes it easy to identify which product categories (e.g., Technology, Furniture, or Office Supplies) contribute the most to the company’s revenue.

**Insights**:

* From the bar plot, it can be observed that **Technology** is likely to be the top-performing category, generating the highest sales, followed by **Furniture** and **Office Supplies**. This suggests that the company may want to focus its efforts on promoting high-performing categories like Technology, while reevaluating its strategy for categories with lower sales.

### **#2: Regional Sales and Profit Performance**

**Objective**:  
The goal of this analysis is to determine which regions contribute the most to total sales and profit. Understanding regional performance is crucial for identifying key markets, directing resources, and strategizing region-specific business operations.

**Code Explanation**:

**1- Grouping Data by Region**:

* + The dataset, after removing outliers, is grouped by the Region column to sum the values of Sales and Profit for each region (West, East, Central, South).
  + The groupby() function is used to aggregate the Sales and Profit values across all regions, allowing us to calculate the total sales and profit contributed by each region.

**Code:**

region\_sales = dataset\_no\_outliers.groupby('Region')['Sales'].sum() region\_profit = dataset\_no\_outliers.groupby('Region')['Profit'].sum()

**Explanation**:  
This step generates two separate Pandas Series, region\_sales and region\_profit, where the index represents each region and the values represent the corresponding total sales and profit for that region. These data are the foundation for the pie charts that follow.

**2- Creating the Pie Charts**:

* We create a figure using plt.figure(figsize=(12, 6)) to define the size of the overall plot. This creates a large enough canvas for displaying two pie charts side by side.

**Code:**

plt.figure(figsize=(12, 6))

**Explanation**:  
By specifying the size, we ensure the charts are visually clear and easy to interpret. In this case, a figure size of 12 by 6 inches is chosen to display two charts without overlap.

**3- Plotting the Sales Pie Chart**:

* The first subplot (plt.subplot(1, 2, 1)) is set up for the sales pie chart. This code tells Python to create a layout with 1 row and 2 columns of subplots, and we're currently working on the first subplot.
* The plt.pie() function is used to create a pie chart of region\_sales, with the region names as labels and the percentage contribution of each region displayed using autopct='%1.1f%%'. The pie chart is rotated to start at a 140-degree angle for better visual alignment, and the colors parameter applies a color palette to differentiate the regions.

**Code:**

plt.subplot(1, 2, 1) plt.pie(region\_sales, labels=region\_sales.index, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors) plt.title('Total Sales by Region')

**Explanation**:  
This pie chart provides a visual representation of the proportion of total sales each region contributes. The labels make it easy to identify the regions, and the percentage values help understand the exact contribution of each region to overall sales. The color scheme enhances clarity and distinction between regions.

**4- Plotting the Profit Pie Chart**:

* The second subplot (plt.subplot(1, 2, 2)) is set up to plot the profit data. The code is nearly identical to the sales pie chart, except it uses region\_profit to visualize the profit distribution across regions.

**Code:**

plt.subplot(1, 2, 2) plt.pie(region\_profit, labels=region\_profit.index, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors) plt.title('Total Profit by Region')

**Explanation**:  
The profit pie chart complements the sales chart, showing the proportion of total profit each region generates. By placing the sales and profit pie charts side by side, we can easily compare the performance of each region in terms of both revenue and profitability.

**Final Adjustments and Display**:

* The plt.tight\_layout() function ensures that the subplots don’t overlap, providing a neat, clean layout. The plt.show() function renders the charts for visualization.

**Code:**

plt.tight\_layout()

plt.show()

**Explanation**:  
This final step adjusts the spacing between the charts to ensure they are well-displayed without crowding. The use of plt.show() ensures that both charts are rendered for immediate analysis.

### **Results and Insights:**

1. **Sales Insights**:
   * From the pie chart, the **West** region emerges as the highest contributor to total sales, followed by the **East** region. The **Central** and **South** regions trail behind in terms of revenue generation, indicating that the **West** and **East** are key markets for the business.
2. **Profit Insights**:
   * The profit distribution shows a similar pattern, with the **West** region leading in profitability. The **East** region also contributes significantly to profits, while the **Central** and **South** regions show lower profit contributions, indicating a potential area for improvement.

**Key Takeaways**:

* The **West** and **East** regions are the most profitable and highest in sales, which suggests these areas should receive continued focus in terms of resources, marketing, and operations.
* **Central** and **South** regions, although smaller contributors, could represent opportunities for growth and optimization, especially in improving profit margins.

These pie charts provide a clear visual breakdown of regional performance, which is vital for making informed decisions about resource allocation and regional strategies.

### **#3: Which Regions Have the Highest Sales and Profit Performance? (Map Chart)**

To analyze sales and profit performance across different regions, we used a map visualization to highlight total sales and profit figures for each region. Below is a breakdown of the process:

**1- Data Aggregation**: The first step involved grouping the dataset by the Region column and calculating the total sales and profit for each region. This was achieved with the following code:

region\_sales\_profit = dataset\_no\_outliers.groupby('Region')[['Sales', 'Profit']].sum().reset\_index()

**2- Region Coordinates**: We defined the geographical coordinates (latitude and longitude) for each region to accurately place markers on the map. The coordinates were assigned as follows:

region\_coords = { 'West': [37.7749, -122.4194], # Example coordinates for regions 'East': [40.7128, -74.0060], 'South': [29.7604, -95.3698], 'Central': [41.8781, -87.6298] }

**3- Map Initialization**: A Folium map centered on the United States was created to visualize the sales and profit data. The zoom level was set to 4 for a suitable view:

us\_map = folium.Map(location=[37.0902, -95.7129], zoom\_start=4)

**4- Marker Clustering**: To manage the markers efficiently, a marker cluster was added to the map. This allows for better visualization of data points that are close together:

marker\_cluster = MarkerCluster().add\_to(us\_map)

**5- Adding Markers**: For each region in the region\_sales\_profit DataFrame, a marker was created and added to the map. The following steps were performed for each region:

* The sales and profit data were retrieved.
* The corresponding latitude and longitude were extracted from region\_coords.
* A popup was created to display the region's sales and profit when the marker is clicked:

popup\_text = f'Region: {region}<br>Sales: ${sales:,.2f}<br>Profit: ${profit:,.2f}'

* A fixed label next to each marker was also added to provide immediate visibility of the sales and profit figures:

label\_text = f'{region}: Sales ${sales:,.2f} | Profit ${profit:,.2f}'

* Finally, the markers were added to the map with the following code:

folium.Marker( location=[lat, lon], popup=popup\_text ).add\_to(marker\_cluster) folium.map.Marker( [lat, lon], icon=folium.DivIcon(html=f""" <div style="font-size: 12px; color: black;"> {label\_text} </div> """) ).add\_to(us\_map)

**6- Map Output**: The final map was saved to an HTML file, allowing for interactive viewing of the sales and profit performance by region:

us\_map.save("sales\_profit\_region\_map.html")

This visualization effectively illustrates which regions have the highest sales and profit performance, allowing stakeholders to easily identify areas of strength and opportunities for improvement.

### **#4: Sales Variation by Shipping Mode**

This analysis aims to explore how total sales differ across various shipping modes. To achieve this, we grouped the sales data based on the 'Ship Mode' variable, summing up the sales for each category. The following steps detail the process:

**1- Data Grouping and Summation**: The dataset is grouped by the 'Ship Mode' column, and the total sales are calculated for each shipping mode. This is done using the groupby() function in conjunction with sum(), which aggregates the sales data, allowing us to assess the performance of each shipping method.

shipping\_mode\_sales = dataset\_no\_outliers.groupby('Ship Mode')['Sales'].sum().sort\_values()

**2- Visualization**: A bar plot is created to visually represent the total sales across the different shipping modes. The plot's dimensions are set to 10 by 6 inches to ensure clarity.

plt.figure(figsize=(10, 6))

**3- Plotting the Data**: The summed sales data is then plotted as a bar chart using the plot() method, with a light coral color chosen for the bars to enhance visibility. The title and labels for the x-axis and y-axis are also added to provide context for the data.

shipping\_mode\_sales.plot(kind='bar', color='lightcoral')

plt.title('Sales Performance Across Different Shipping Modes')

plt.xlabel('Shipping Mode')

plt.ylabel('Total Sales')

**4- Axis Customization**: The x-ticks are rotated at a 45-degree angle for better readability, especially if shipping mode names are lengthy. A grid is also added along the y-axis with dashed lines for easier interpretation of the sales values.

plt.xticks(rotation=45)

plt.grid(axis='y', linestyle='--', alpha=0.6)

**5- Displaying the Plot**: Finally, the show() method is called to render the bar plot, allowing for immediate visualization of the data.

plt.show()

The resulting bar plot effectively illustrates the variation in total sales attributed to different shipping modes, providing valuable insights into which shipping methods are most profitable.

### **#5: Sales Performance Over Time**

To evaluate the sales performance over time, we analyze the trends on a monthly, quarterly, and yearly basis using the sales data in the dataset. The following steps were taken to derive these insights:

**1- Conversion of 'Order Date' to Datetime Format**: The 'Order Date' column is converted to a datetime format to facilitate time-based analysis. The conversion is performed using the pd.to\_datetime() function, with errors='coerce' to handle any invalid date formats by converting them to NaT (Not a Time). This ensures that the dates are in a usable format for further analysis.

dataset\_no\_outliers.loc['Order Date'] = pd.to\_datetime(dataset\_no\_outliers['Order Date'], errors='coerce')

**2- Verification of Conversion**: We confirm the successful conversion by displaying the first few entries of the 'Order Date' column. This step ensures that the dates are correctly formatted and ready for extraction of time components.

print(dataset\_no\_outliers['Order Date'].head())

**3- Extraction of Time Components**: The month, quarter, and year are extracted from the 'Order Date' for analysis:

* **Monthly**: The month is derived using dt.to\_period('M'), allowing us to group sales by month.
* **Quarterly**: The quarter is derived using dt.to\_period('Q'), facilitating quarterly sales analysis.
* **Yearly**: The year is directly extracted using dt.year.

dataset\_no\_outliers.loc[:, 'Month'] = dataset\_no\_outliers['Order Date'].dt.to\_period('M') dataset\_no\_outliers.loc[:, 'Quarter'] = dataset\_no\_outliers['Order Date'].dt.to\_period('Q') dataset\_no\_outliers.loc[:, 'Year'] = dataset\_no\_outliers['Order Date'].dt.year

**4- Grouping Sales Data**: The sales data is grouped by the extracted time components to calculate total sales:

* **Monthly Sales**: Sales are summed for each month.
* **Quarterly Sales**: Sales are summed for each quarter.
* **Yearly Sales**: Sales are summed for each year.

monthly\_sales = dataset\_no\_outliers.groupby('Month')['Sales'].sum() quarterly\_sales = dataset\_no\_outliers.groupby('Quarter')['Sales'].sum() yearly\_sales = dataset\_no\_outliers.groupby('Year')['Sales'].sum()

**5- Visualization of Sales Trends**: The sales performance is visualized using line plots for better insight into trends over time. The following plots are generated:

* **Monthly Sales Trend**: A line plot depicting the total sales per month, with markers for each data point.

plt.figure(figsize=(10, 6))

monthly\_sales.plot(kind='line', marker='o', color='b')

plt.title('Monthly Sales Trend')

plt.xlabel('Month')

plt.ylabel('Total Sales')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

* **Quarterly Sales Trend**: A line plot showing the total sales per quarter.

plt.figure(figsize=(10, 6))

quarterly\_sales.plot(kind='line', marker='o', color='g')

plt.title('Quarterly Sales Trend')

plt.xlabel('Quarter')

plt.ylabel('Total Sales')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

* **Yearly Sales Trend**: A line plot illustrating the total sales per year.

plt.figure(figsize=(10, 6))

yearly\_sales.plot(kind='line', marker='o', color='r')

plt.title('Yearly Sales Trend')

plt.xlabel('Year')

plt.ylabel('Total Sales')

plt.tight\_layout()

plt.show()

These visualizations allow for a comprehensive understanding of how sales performance evolves over time, revealing patterns and trends that are essential for strategic planning and decision-making.

### **#6: Which Customer Segments Generate the Highest Sales?**

To analyze the sales performance across different customer segments, we created a scatter plot combined with a line chart. This visualization allows us to easily identify which segments contribute most significantly to total sales. Below is a breakdown of the code used to generate this insight:

# Group the data by 'Segment' and calculate total sales

segment\_sales = dataset\_no\_outliers.groupby('Segment')['Sales'].sum()

In this first step, we group the dataset by the 'Segment' column and calculate the total sales for each segment. The result, segment\_sales, is a Pandas Series containing the total sales figures for each customer segment.

# Create the figure and axis

plt.figure(figsize=(10, 6))

We initialize a new figure for the plot with a specified size of 10x6 inches, which provides ample space for the data points and labels.

# Plot the scatter plot

plt.scatter(segment\_sales.index, segment\_sales, color='purple', s=100, label='Sales by Segment')

A scatter plot is created to represent the total sales for each customer segment. The segments are represented on the x-axis, while the corresponding total sales are on the y-axis. The data points are colored purple and are set to a size of 100 for better visibility.

# Plot the line connecting the points

plt.plot(segment\_sales.index, segment\_sales, color='blue', linestyle='-', linewidth=2, label='Trendline')

A line is plotted to connect the data points, providing a visual representation of the trend across customer segments. The line is styled with a solid blue color and a width of 2, enhancing the overall clarity of the plot.

# Annotate each point with the sales value

for i, value in enumerate(segment\_sales):

plt.text(segment\_sales.index[i], value, f'${value:.2f}', ha='center', va='bottom', fontsize=12)

Each point on the scatter plot is annotated with its corresponding sales value, formatted to two decimal places. The annotations are positioned above each point to avoid overlapping, making it easier for viewers to interpret the data.

# Add title and labels

plt.title('Total Sales Across Customer Segments (Scatter and Line Plot)')

plt.xlabel('Customer Segment')

plt.ylabel('Total Sales')

plt.grid(True, linestyle='--', alpha=0.7)

We enhance the plot’s readability by adding a title and labeling the x-axis as 'Customer Segment' and the y-axis as 'Total Sales'. A grid is also included to aid in visual alignment, styled with a dashed line and set to 70% transparency.

# Add a legend to explain the plot elements

plt.legend()

A legend is included to clarify the meaning of the plot elements, indicating which color represents the sales data and which represents the trendline.

# Display the plot

plt.show()

Finally, the plot is displayed, allowing for a visual assessment of which customer segments generate the highest sales.

### **#7: Average Order Value (AOV) Across Different Regions and Customer Segments**

This analysis aims to identify which regions have the highest average order value (AOV) across various customer segments. The following steps were performed to achieve this:

**1- Sales and Order Calculation**:

* + We first calculated the total sales and the total number of unique orders for each combination of region and customer segment. This was accomplished using the groupby function on the dataset\_no\_outliers, which allowed us to aggregate the sales and order counts efficiently.

**Code:**

region\_segment\_sales = dataset\_no\_outliers.groupby(['Region', 'Segment'])['Sales'].sum().reset\_index()

region\_segment\_orders = dataset\_no\_outliers.groupby(['Region', 'Segment'])['Order ID'].nunique().reset\_index()

**2- Data Merging**:

* The two resulting DataFrames, region\_segment\_sales and region\_segment\_orders, were merged based on the Region and Segment columns. This step is essential to combine sales and order information into a single DataFrame for further analysis.

**Code:**

region\_segment\_data = pd.merge(region\_segment\_sales, region\_segment\_orders, on=['Region', 'Segment'])

region\_segment\_data.rename(columns={'Sales': 'Total Sales', 'Order ID': 'Total Orders'}, inplace=True)

**3- Average Order Value (AOV) Calculation**:

* A new column, AOV, was created by dividing the total sales by the total number of orders for each region and customer segment. This provides a clear view of the average revenue generated per order.

**Code:**

region\_segment\_data['AOV'] = region\_segment\_data['Total Sales'] / region\_segment\_data['Total Orders']

**4- Pivot Table Creation**:

* A pivot table was generated from the region\_segment\_data to visualize the AOV across different regions and customer segments. This pivot table organized the data, allowing for easy comparison across segments.

**Code:**

pivot\_aov = region\_segment\_data.pivot\_table(index='Region', columns='Segment', values='AOV', fill\_value=0)

**5- Heatmap Visualization**:

* A heatmap was created to visually represent the AOV across different regions and customer segments. The heatmap uses a color gradient to indicate the magnitude of the AOV values, with annotations for clarity.

**Code:**

plt.figure(figsize=(12, 8))

sns.heatmap(pivot\_aov, annot=True, cmap='YlGnBu', fmt='.2f', linewidths=0.5) plt.title('Average Order Value (AOV) Across Different Regions and Customer Segments') plt.xlabel('Customer Segment')

plt.ylabel('Region')

plt.show()

**6- Bar Chart Visualization**:

* Additionally, a bar chart was generated to compare the AOV across regions and customer segments. This chart provides an alternative view, highlighting differences in AOV more distinctly.

**Code:**

plt.figure(figsize=(14, 8))

sns.barplot(x='Region', y='AOV', hue='Segment', data=region\_segment\_data, palette='muted')

plt.title('Average Order Value (AOV) Comparison Across Regions and Customer Segments') plt.xlabel('Region')

plt.ylabel('Average Order Value (AOV)')

plt.legend(title='Customer Segment', bbox\_to\_anchor=(1.05, 1), loc='upper left') plt.grid(axis='y', linestyle='--', alpha=0.6)

plt.tight\_layout()

plt.show()

This structured explanation covers each significant step in the code and provides context for the calculations and visualizations created to analyze the AOV across different regions and customer segments.

### **#8: Which Shipping Mode Yields the Highest Profit Margins?**

This analysis aims to determine which shipping mode offers the highest profit margins by calculating and visualizing the profit margins for each shipping mode in our dataset.

**1- Data Aggregation**: The first step involves grouping the dataset by the shipping mode to calculate total profit and sales. This is accomplished using the groupby method, which aggregates the Profit and Sales columns for each shipping mode:

shipping\_mode\_profit = dataset\_no\_outliers.groupby('Ship Mode')[['Profit', 'Sales']].sum().reset\_index()

The reset\_index() function is used to convert the resulting grouped data back into a DataFrame format.

**2- Profit Margin Calculation**: Next, we compute the profit margin for each shipping mode. The profit margin is calculated as a percentage using the formula:  
Profit Margin (%)=(ProfitSales)×100\text{Profit Margin (\%)} = \left( \frac{\text{Profit}}{\text{Sales}} \right) \times 100Profit Margin (%)=(SalesProfit​)×100  
This calculation is performed for each shipping mode and added as a new column to the shipping\_mode\_profit DataFrame:

shipping\_mode\_profit['Profit Margin (%)'] = (shipping\_mode\_profit['Profit'] / shipping\_mode\_profit['Sales']) \* 100

**3- Data Visualization**: To visualize the profit margins for each shipping mode, a bar chart is created using Matplotlib and Seaborn:

plt.figure(figsize=(10, 6))

sns.barplot(x='Ship Mode', y='Profit Margin (%)', data=shipping\_mode\_profit, palette='viridis')

plt.title('Profit Margins for Different Shipping Modes')

plt.xlabel('Shipping Mode')

plt.ylabel('Profit Margin (%)')

plt.grid(axis='y', linestyle='--', alpha=0.7)

* The figure() function sets the size of the plot.
* The barplot() function generates the bar chart, with shipping modes on the x-axis and profit margins on the y-axis, using a color palette called 'viridis'.
* Titles and labels for the axes are added for clarity.

**4- Annotation**: To enhance the chart's readability, profit margin values are displayed on top of each bar. This is achieved with a loop that iterates through each row of the shipping\_mode\_profit DataFrame:

for index, row in shipping\_mode\_profit.iterrows():

plt.text(index, row['Profit Margin (%)'] + 0.5, f'{row["Profit Margin (%)"]:.2f}%', ha='center', fontsize=12)

The plt.text() function places the calculated profit margin as text above each bar, with some vertical offset for better visibility.

**5- Display**: Finally, the completed chart is displayed using plt.show().

This analysis and visualization provide valuable insights into how different shipping modes perform in terms of profit margins, allowing stakeholders to make informed decisions regarding shipping strategies.

### **#9: Top Products in Terms of Sales and Profit Margins**

To identify the top products regarding both sales and profit margins, we first aggregated the dataset by Product ID. The total sales and total profit for each product were computed, followed by the calculation of the profit margin expressed as a percentage. The resulting data was then sorted to determine the top 10 products based on sales volume.

**Code:**

# Group by 'Product ID' and calculate total sales and total profit

product\_performance = dataset\_no\_outliers.groupby('Product ID')[['Sales', 'Profit']].sum().reset\_index()

# Calculate profit margin for each product

product\_performance['Profit Margin (%)'] = (product\_performance['Profit'] / product\_performance['Sales']) \* 100

# Sort the products by sales and get the top 10

top\_products = product\_performance.sort\_values(by='Sales', ascending=False).head(10)

Next, we visualized the findings using a bar chart, depicting both sales figures and profit margins for these top products.

The visualization was designed with dual axes to clearly differentiate between the two metrics. The primary y-axis represents sales (in dollars), illustrated with a blue bar chart, while the secondary y-axis displays profit margins (in percentage), represented by a line chart. This dual representation allows for an intuitive understanding of how sales performance correlates with profitability.

**Code:**

# Visualization using a bar chart for both Sales and Profit Margin

plt.figure(figsize=(14, 8))

# Create a twin axis to plot sales and profit margin

ax1 = plt.subplot(111)

sns.barplot(x='Product ID', y='Sales', data=top\_products, ax=ax1, color='skyblue', label='Sales')

ax1.set\_ylabel('Sales ($)', color='skyblue')

ax1.tick\_params(axis='y', labelcolor='skyblue')

ax1.set\_xticklabels(top\_products['Product ID'], rotation=45, ha='right')

# Create a second y-axis for profit margin

ax2 = ax1.twinx()

sns.lineplot(x='Product ID', y='Profit Margin (%)', data=top\_products, ax=ax2, color='orange', marker='o', label='Profit Margin')

ax2.set\_ylabel('Profit Margin (%)', color='orange')

ax2.tick\_params(axis='y', labelcolor='orange')

# Title and grid

plt.title('Top 10 Products by Sales and Their Profit Margins')

ax1.grid(axis='y', linestyle='--', alpha=0.7)

# Show legends

ax1.legend(loc='upper left')

ax2.legend(loc='upper right')

plt.tight\_layout()

plt.show()

In summary, this analysis reveals the products that contribute most significantly to overall profitability by examining both their sales performance and profit margins. The visual representation aids in discerning the relationship between high sales and their corresponding profit margins, providing valuable insights for strategic decision-making in product management.

### **#10: Distribution of High-Value Orders Across Product Categories and Regions**

This analysis aims to understand the distribution of high-value orders across different product categories and regions. High-value orders are defined as those with sales exceeding a specified threshold, which in this case is set to **500**. This threshold can be adjusted based on business requirements.

**Code Explanation:**

**1- Define the High-Value Order Threshold:**

high\_value\_threshold = 500 # You can change this value as needed

The threshold for classifying an order as high-value is set to **500**. This value can be modified based on specific analysis needs.

**2- Filter the Dataset for High-Value Orders:**

high\_value\_orders = dataset\_no\_outliers[dataset\_no\_outliers['Sales'] > high\_value\_threshold]

Here, the dataset is filtered to include only those orders where the sales amount exceeds the defined threshold. This filtering is applied to a pre-processed dataset, dataset\_no\_outliers, ensuring that the analysis is free from outlier influence.

**3- Group by 'Category' and 'Region' to Count High-Value Orders:**

high\_value\_distribution = high\_value\_orders.groupby(['Category', 'Region']).size().unstack(fill\_value=0)

The filtered high-value orders are then grouped by **Category** and **Region**. The size() function counts the number of occurrences for each combination of category and region. The unstack() function reshapes the resulting series into a DataFrame format, filling any missing values with **0**.

**4- Visualization Using a Stacked Bar Chart:**

plt.figure(figsize=(14, 8)) high\_value\_distribution.plot(kind='bar', stacked=True, colormap='viridis', ax=plt.gca())

A stacked bar chart is created to visualize the distribution of high-value orders. Each bar represents a product category, while the segments of the bar show the number of high-value orders from different regions. The viridis colormap is used to provide a clear and visually appealing representation.

**5- Adding Chart Elements:**

plt.title('Distribution of High-Value Orders Across Product Categories and Regions') plt.xlabel('Product Category')

plt.ylabel('Number of High-Value Orders')

plt.xticks(rotation=45)

plt.legend(title='Region', bbox\_to\_anchor=(1.05, 1), loc='upper left')

plt.grid(axis='y', linestyle='--', alpha=0.7)

The chart is enhanced with a title, axis labels, and a legend to clearly convey the information. The x-ticks are rotated for better readability, and a grid is added to the y-axis to facilitate easier interpretation of values.

**6- Display the Plot:**

plt.tight\_layout()

plt.show()

Finally, the layout is adjusted for optimal presentation, and the chart is displayed.

This analysis provides insights into which product categories and regions are generating high-value orders, helping to inform strategic decisions regarding marketing and inventory management.

#### **3.4 Forecasting with Python**

Using machine learning, we built a linear regression model to forecast future sales.

**Steps Involved**:

**1- Data Preparation**:

* + We first converted the Order Date to a datetime format and extracted the month as a new feature for time-series forecasting.

**Code**:

df['Order Date'] = pd.to\_datetime(df['Order Date'])

df['Month'] = df['Order Date'].dt.month

**2- Train-Test Split**:

* The dataset was split into training and testing sets to ensure our model could generalize well to unseen data.

**Code**:

from sklearn.model\_selection import train\_test\_split

X = df[['Month']]

y = df['Sales']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2)

**3- Linear Regression Model**:

* We used the LinearRegression model from scikit-learn to predict future sales based on past sales data.

**Code**:

from sklearn.linear\_model import LinearRegression

model = LinearRegression()

model.fit(X\_train, y\_train)

**4- Future Sales Forecast**:

* We used the trained model to predict sales for the next 12 months and plotted the results against historical sales to visualize future trends.

**Code**:

future\_sales = model.predict(X\_test)

**Explanation**:

The forecasted sales provide actionable insights for the next 12 months, helping management to make data-driven decisions about inventory, marketing, and staffing.

### **3.5 Tableau Dashboards: Detailed Explanation**

The final step in our analysis involved using **Tableau** to create four highly interactive dashboards. These dashboards were designed to provide a user-friendly way to explore the insights gained from the Python analysis and make it easier for stakeholders to digest the findings and apply them to decision-making. Below is a comprehensive breakdown of each Tableau dashboard and how the visual elements contribute to the overall analysis.

#### **Dashboard 1: Sales Overview**

**Objective**:  
To provide an at-a-glance view of the overall business performance, including total sales, profits, and product quantities sold across different regions and product categories.

**Key Components**:

1. **Regional Sales and Profit Distribution**:
   * **Map Visualization**: A map of the United States was used to represent total sales and profit across different regions (West, East, Central, South). Each region was color-coded based on performance, with darker shades representing higher values.
   * **Interactivity**: Users can hover over or click on any region to see detailed breakdowns of sales, profit, and units sold.
   * **Insight**: This visualization helps identify which regions contribute the most to overall revenue and profit. For instance, the **West** was shown to be the most profitable, while the **Central** region lagged behind.
2. **Category Breakdown**:
   * **Bar Chart**: A bar chart was used to display the total sales for each product category (Furniture, Office Supplies, and Technology).
   * **Interactive Filters**: Filters for region and customer segment allow the user to see how product categories perform in specific regions or for particular customer types (e.g., Corporate, Consumer, or Home Office).
   * **Insight**: This analysis reveals that while **Office Supplies** had the largest volume of sales, **Technology** had higher profit margins. Businesses can use this insight to focus on promoting high-margin products.
3. **Sales, Profit, and Quantity KPIs**:
   * **Key Performance Indicators (KPIs)** were displayed at the top of the dashboard, providing a snapshot of the most important metrics: total sales, total profit, and total quantity sold. This high-level summary helps executives get a quick sense of overall performance.
   * **Insight**: These KPIs allow stakeholders to understand, at a glance, the success of the store in terms of revenue generation and profitability.

**Recommendation**:  
This dashboard is best suited for high-level decision-makers who need a quick overview of sales performance and profitability across regions and categories. The interactive elements enable deeper dives into specific areas of interest, allowing stakeholders to tailor their analysis to particular segments of the business.

#### **Dashboard 2: Profit Analysis**

**Objective**:  
To provide a detailed analysis of profit generation across different dimensions, including customer segments, product categories, and shipping modes. This dashboard is tailored to uncover which areas of the business are contributing the most to overall profitability.

**Key Components**:

1. **Profit by Customer Segment**:
   * **Stacked Bar Chart**: This chart visualizes profit contributions from different customer segments (Consumer, Corporate, and Home Office) across various regions. Each bar represents a region, and segments are color-coded within the bars.
   * **Interactivity**: Filters allow users to toggle between different product categories to see how each category performs across customer segments and regions.
   * **Insight**: This analysis highlights that **Corporate** customers typically have higher profit margins than **Consumer** and **Home Office** segments, particularly in the **West** region.
2. **Shipping Mode vs. Profit**:
   * **Bar and Line Chart**: This dual-axis chart combines a bar chart (total sales by shipping mode) with a line plot (profit margin for each shipping mode). The goal is to see which shipping modes drive the most revenue while maintaining strong profit margins.
   * **Insight**: The analysis revealed that **Standard Class** was the most frequently used shipping mode, but **First Class** and **Same Day** shipping had higher profit margins. This insight suggests an opportunity to encourage customers to choose faster, more profitable shipping options.
3. **Profit by Product Category**:
   * **Pie Chart**: A pie chart was used to display the proportion of profit generated by different product categories. The chart shows that **Technology** contributes disproportionately to overall profits despite not having the highest sales volume.
   * **Interactive Drill-Down**: Users can click on specific categories to drill down into subcategories (e.g., comparing sales of laptops vs. peripherals within Technology) to gain a deeper understanding of what is driving profit within each category.
   * **Insight**: This insight is useful for focusing marketing efforts on high-margin products like **Technology**, while reevaluating pricing strategies for lower-margin categories like **Furniture**.

**Recommendation**:  
This dashboard is ideal for finance teams or product managers who need to understand profit generation in detail. The comparison between shipping modes and customer segments allows the business to identify the most profitable combinations and optimize their operations accordingly.

#### **Dashboard 3: Discount & Trend Analysis**

**Objective**:  
To analyze how discounts impact sales, profit, and customer behavior over time, and to visualize key sales trends at a monthly, quarterly, and yearly level. This dashboard is particularly useful for marketing and sales teams looking to optimize promotions and campaigns.

**Key Components**:

1. **Discount Impact on Profit**:
   * **Scatter Plot**: This scatter plot compares the level of discount offered to the resulting profit. Each point represents an individual sale, and the color-coding shows different customer segments.
   * **Trend Line**: A regression trend line was added to show the general relationship between discount size and profit, with a clear downward trend indicating that higher discounts often lead to reduced profits.
   * **Insight**: This plot reveals a significant correlation between higher discounts and lower profit margins. Although discounts can increase sales volume, they can negatively affect overall profitability.
2. **Sales Trends Over Time**:
   * **Line Chart**: This line chart displays monthly and quarterly sales trends over the years, helping to identify seasonal patterns and high-demand periods.
   * **Interactive Time Filters**: Users can filter the data by year or month to see how sales fluctuated during different time periods (e.g., during holiday seasons or back-to-school months).
   * **Insight**: The line chart shows a clear sales spike in the fourth quarter, likely due to holiday shopping. This indicates that marketing and inventory strategies should focus on Q4 to maximize sales and profit.
3. **Sales vs. Discounts by Category**:
   * **Bar Chart**: This chart shows the total sales for each product category, alongside the average discount offered for each category. The bar chart helps to visualize whether specific categories (e.g., Furniture) rely more heavily on discounts to drive sales.
   * **Insight**: The analysis shows that **Furniture** products typically involve higher discounts than other categories, suggesting that customers are more price-sensitive in this category. In contrast, **Technology** products have lower discounts but generate higher profits, indicating they can command higher prices.

**Recommendation**:  
This dashboard is perfect for marketing and pricing teams looking to balance promotions and profit. It provides valuable insights into which discount strategies are effective for driving sales without sacrificing profitability, and helps identify optimal times for promotional campaigns.

#### **Dashboard 4: Sales Forecasting**

**Objective**:  
To visualize the results of the sales forecasting model developed in Python. This dashboard combines historical sales data with forecasted sales for the next 12 months, allowing stakeholders to anticipate future performance and make data-driven decisions for inventory and marketing planning.

**Key Components**:

1. **Historical vs. Forecasted Sales**:
   * **Line Chart**: A line chart was used to visualize both historical sales (actual data) and future forecasted sales (predicted data). Historical data is shown in one color, while forecasted data is displayed in a different shade, clearly distinguishing the two.
   * **Interactivity**: Filters for region, product category, and customer segment allow users to refine the forecast for specific areas of interest (e.g., forecasting sales for **Technology** products in the **West** region).
   * **Insight**: The forecast reveals an overall upward trend in sales for the next 12 months, suggesting steady growth. The model also identified seasonal fluctuations, with expected sales peaks in Q4 and slight declines in the mid-year period.
2. **Predicted Sales by Region**:
   * **Bar Chart**: A bar chart shows the forecasted sales for each region, broken down by quarter. This visualization helps stakeholders plan region-specific strategies for the upcoming year.
   * **Insight**: The **West** and **East** regions are expected to continue their dominance in terms of sales, but the forecast predicts significant growth in the **South**, suggesting untapped potential in that region.
3. **Forecast Accuracy Evaluation**:
   * **Gauge Chart**: A gauge chart was included to show the accuracy of the forecast model by comparing predicted sales with actual historical sales for the last 6 months. This chart helps stakeholders evaluate the reliability of the forecasting model.
   * **Insight**: The gauge chart shows that the model performed well, with predictions falling within an acceptable margin of error. This increases confidence in using the model for future business decisions.

**Recommendation**:  
This dashboard is highly valuable for strategic planners and inventory managers. By providing accurate sales forecasts, it helps the business make informed decisions about production, staffing, and marketing budgets. The regional breakdown allows for focused planning, ensuring resources are allocated efficiently.

### **Conclusion: Tableau Dashboards as a Decision-Making Tool**

The Tableau dashboards created for this project transform the raw insights generated from Python analysis into interactive, visually compelling tools for decision-making. Each dashboard serves a distinct purpose, from high-level overviews to deep dives into profit, discount, and forecast analysis, making them accessible to a wide range of stakeholders. Whether for executives looking at top-line numbers or for marketing teams planning their next promotional strategy, these dashboards provide the clarity and insights necessary for data-driven decisions that will drive future success for the Superstore.

### **4. Key Insights and Recommendations**

1. **Top-Performing Regions**:
   * The **West** region generated the highest sales and profit. Businesses should focus on expanding in this region, while exploring marketing strategies to improve performance in the underperforming **Central** and **South** regions.
2. **Product Category Insights**:
   * **Technology** products had the highest profit margins, whereas **Office Supplies** contributed the most to total sales. Increasing marketing for high-margin products like technology could improve overall profitability.
3. **Shipping Mode Optimization**:
   * **Standard Class** shipping accounted for the majority of sales. Encouraging customers to use **First Class** or **Same Day** shipping (which had higher profit margins) can enhance profitability.
4. **Sales Trends**:
   * The last quarter of the year saw the highest sales, likely due to holiday shopping. Businesses should capitalize on these trends by ramping up promotions and stock during Q4.

### **5. Conclusion**

By integrating Python and Tableau, we were able to extract meaningful insights from the Superstore data. This project demonstrates the power of data-driven decision-making, offering valuable insights into product, customer, and regional performance. The forecasting model adds an additional layer of strategic planning, helping the business to prepare for future opportunities and challenges.