DATA ANALYTICS WITH COGNOS

PHASE-5

PROJECT TITLE:

PRODUCT SALES ANALYSIS

OBJECTIVE:

The objective of this project is to load, preprocess, analyze, and visualize a dataset using IBM Cognos, ultimately creating a comprehensive document for assessment, showcasing the insights and findings derived from the data analysis. Sales objectives for selling products could include: Increasing the size of average deals. Increasing annual up-sells. Increasing quarterly cross-sells.

1. Data Import and Preparation:

- Import the dataset from Kaggle into IBM Cognos.

- Clean and preprocess the data, handling missing values, and data formatting.

2. Product Development Analysis:

- Define your objectives for product development based on the dataset.

- Explore the product sales data to identify trends, popular products, and market demand.

- Develop insights into potential new products or improvements to existing products.

3. Sale Analysis:

- Define the scale you want to analyze (e.g., scaling up sales, expanding product lines, or market reach).

- Analyze the dataset to identify areas for scaling, such as regions or products with the highest growth potential.

4. Statistical Analysis and Visualization:

- Use tools in Watson Studio (e.g., Python with libraries like Pandas, Matplotlib, and Seaborn) to perform statistical analysis and create visualizations.

5. Machine Learning (Optional):

- If applicable, you can apply machine learning algorithms to predict sales trends or customer behavior.

6. Report and Presentation:

- Create a report or presentation summarizing your product development and scale analysis findings.

Key Features:

Data Integration and Cleaning: Use IBM DataStage or IBM InfoSphere to integrate data from various sources such as sales transactions, customer databases, and marketing campaigns. Apply data cleaning techniques to ensure data accuracy.

Data Warehousing: Store the integrated and cleaned data in an IBM Db2 Warehouse, which is optimized for analytics workloads.

AI-Driven Predictive Analytics: Utilize IBM Watson Studio for predictive analytics. Develop machine learning models to forecast sales trends, identify customer behaviors, and recommend pricing strategies.

Natural Language Processing (NLP): Implement IBM Watson Natural Language Understanding to analyze customer feedback, reviews, and social media sentiment to gain insights into product perception and customer satisfaction.

Data Visualization: Create interactive dashboards using IBM Cognos Analytics or IBM Watson Studio's Data Refinery to visualize sales data, trends, and predictions. These dashboards can be customized for different user roles, such as sales managers, marketing teams, and executives.

Recommendation Engine: Build a recommendation engine using IBM Watson Personalization to suggest product bundles or upsell opportunities to customers based on their purchase history and behavior.

Sales Performance Monitoring: Implement real-time sales performance monitoring using IBM Streams. This feature allows businesses to react quickly to changes in sales patterns and adjust strategies accordingly.

Cost Optimization: Utilize IBM Decision Optimization to optimize supply chain and inventory management, helping businesses reduce costs and enhance profitability.

Mobile Access: Develop a mobile app using IBM Mobile Foundation to provide on-the-go access to sales insights and alerts for sales teams and executives.

DATA SOURCE:

		Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4
2	0	13/6/2010	5422	3725	576	907	17187.74	23616.5	3121.92	6466.91
3	1	14/6/2010	7047	779	3578	1574	22338.99	4938.86	19392.76	11222.62
4	2	15/6/2010	1572	2082	595	1145	4983.24	13199.88	3224.9	8163.85
5	3	16/6/2010	5657	2399	3140	1672	17932.69	15209.66	17018.8	11921.36
6	4	17/6/2010	3668	3207	2184	708	11627.56	20332.38	11837.28	5048.04
7	5	18/6/2010	2898	2539	311	1513	9186.66	16097.26	1685.62	10787.69
8	6	19/6/2010	6912	1470	1576	1608	21911.04	9319.8	8541.92	11465.04
9	7	20/6/2010	5209	2550	3415	842	16512.53	16167	18509.3	6003.46
10	8	21/6/2010	6322	852	3646	1377	20040.74	5401.68	19761.32	9818.01
11	9	22/6/2010	6865	414	3902	562	21762.05	2624.76	21148.84	4007.06
12	10	23/6/2010	1287	3955	2710	1804	4079.79	25074.7	14688.2	12862.52
13	11	24/6/2010	2197	1429	2754	1299	6964.49	9059.86	14926.68	9261.87
14	12	25/6/2010	7910	1622	5574	306	25074.7	10283.48	30211.08	2181.78
15	13	26/6/2010	3855	1015	1746	608	12220.35	6435.1	9463.32	4335.04
16	14	27/6/2010	5988	3288	916	1530	18981.96	20845.92	4964.72	10908.9
17	15	28/6/2010	2653	1544	3867	652	8410.01	9788.96	20959.14	4648.76
18	16	29/6/2010	3664	2294	3244	897	11614.88	14543.96	17582.48	6395.61
19	17	30/6/2010	7077	2297	5376	1130	22434.09	14562.98	29137.92	8056.9
20	18	1/7/2010	3509	700	1175	1205	11123.53	4438	6368.5	8591.65
21	19	2/7/2010	3716	3175	651	1263	11779.72	20129.5	3528.42	9005.19
22	20	3/7/2010	7746	2883	671	728	24554.82	18278.22	3636.82	5190.64
23	21	4/7/2010	7006	2833	758	1005	22209.02	17961.22	4108.36	7165.65
24	22	5/7/2010	5223	1923	1583	1877	16556.91	12191.82	8579.86	13383.01
25	23	6/7/2010	4753	3125	2787	583	15067.01	19812.5	15105.54	4156.79
26	24	7/7/2010	3369	752	5913	358	10679.73	4767.68	32048.46	2552.54

DatasetLink: **https://www.kaggle.com/datasets/ksabishek/product-sales-data**

DATA PREPROCESSING:

- Clean the dataset: Check for missing values and outliers.
- Convert the 'Date' column to a date time format for time series analysis.
-Create new features if needed, such as total sales, profit, or seasonality indicators.

1. Import Necessary Libraries:

We start by importing the required Python libraries: Pandas for data manipulation and Matplotlib for data visualization.

2. Read the Dataset:

We read the dataset from a CSV file. You should replace `'your_dataset.csv'` with the actual file path where your dataset is located.

3. Handling Missing or Invalid Dates:

The code drops rows with missing or invalid date values using `data.dropna(subset=['Date'])`. If there are missing or invalid dates, this step ensures the dataset only contains valid date entries.

4.Customization:

You can modify these visualizations by selecting different columns or customizing the plots further. For more complex visualizations or additional analysis, you may need to explore other plotting libraries or techniques, but this code serves as a good starting point for basic data exploration and visualization.

Program:

```
import pandas as pd
import matplotlib.pyplot as plt
# Read the dataset into a Pandas DataFrame
```

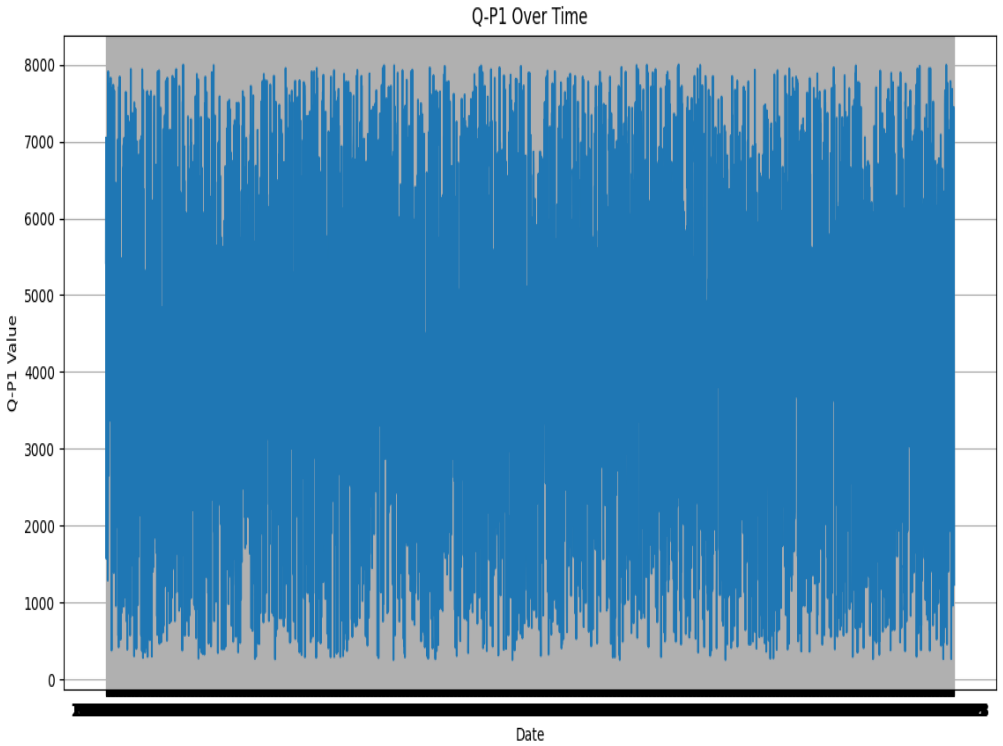
```
data = pd.read_csv('statsfinal.csv')  # Replace 'your_dataset.csv' with the actual file path if the data is in a CSV file
# Fill or drop any missing or invalid date values if needed
data = data.dropna(subset=['Date'])
print(data.info())
print(data.describe())
print(data.head())
# Visualization 1: Line plot of one of the numeric columns (e.g., Q-P1)
plt.figure(figsize=(12, 6))
plt.plot(data['Date'], data['Q-P1'])
plt.title('Q-P1 Over Time')
plt.xlabel('Date')
plt.ylabel('Q-P1 Value')
plt.grid(True)
plt.show()
# Visualization 2: Scatter plot between two numeric columns (e.g., Q-P1 vs. S-P1)
plt.figure(figsize=(10, 8))
plt.scatter(data['Q-P1'], data['S-P1'], alpha=0.5)
plt.title('Scatter Plot: Q-P1 vs. S-P1')
plt.xlabel('Q-P1')
plt.ylabel('S-P1')
plt.grid(True)
plt.show()
# Visualization 3: Histogram of a numeric column (e.g., Q-P1)
plt.figure(figsize=(10, 6))
plt.hist(data['Q-P1'], bins=20, edgecolor='k')
plt.title('Histogram of Q-P1')
plt.xlabel('Q-P1 Value')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

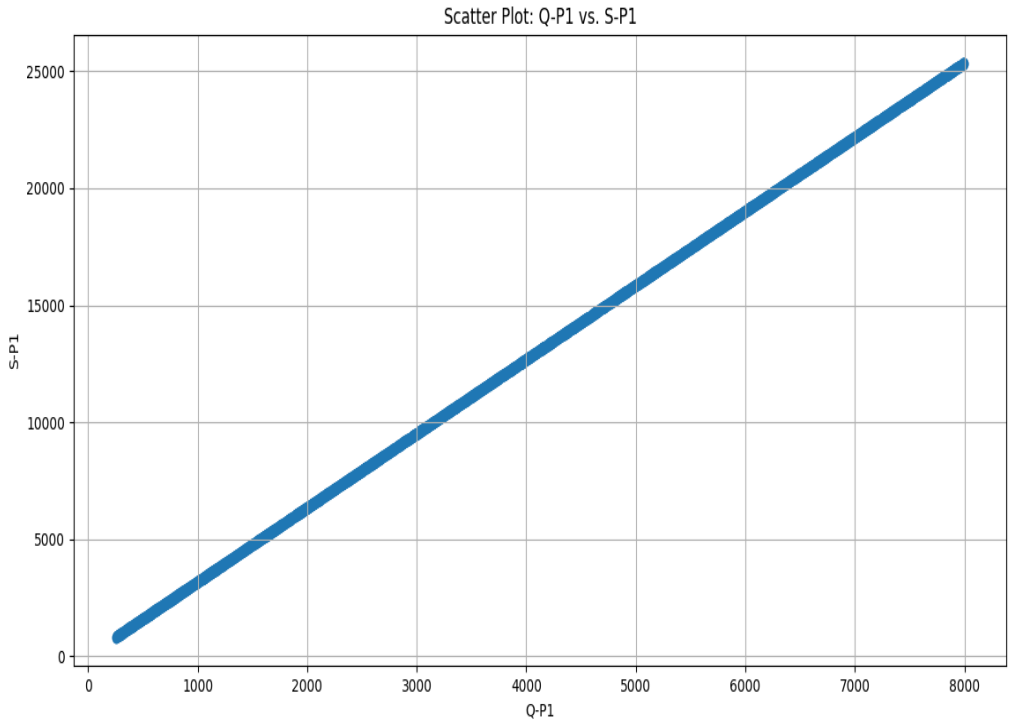
Output:

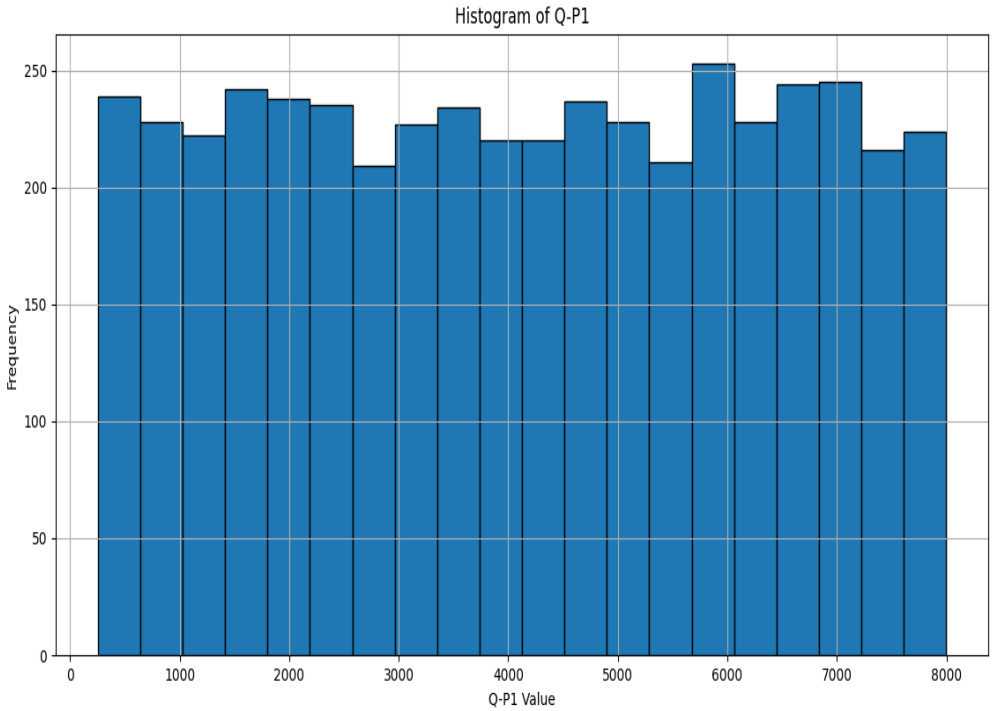
```
IDLE Shell 3.11.2
File  Edit  Shell  Debug  Options  Window  Help
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: D:/aaa.py ==========
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Unnamed: 0    4600 non-null   int64
 1   Date          4600 non-null   object
 2   Q-P1          4600 non-null   int64
 3   Q-P2          4600 non-null   int64
 4   Q-P3          4600 non-null   int64
 5   Q-P4          4600 non-null   int64
 6   S-P1          4600 non-null   float64
 7   S-P2          4600 non-null   float64
 8   S-P3          4600 non-null   float64
 9   S-P4          4600 non-null   float64
dtypes: float64(4), int64(5), object(1)
memory usage: 359.5+ KB
None
         Unnamed: 0         Q-P1  ...         S-P3         S-P4
count   4600.000000  4600.000000  ...  4600.000000  4600.000000
mean    2299.500000  4121.849130  ...  17049.910800  8010.555000
std     1328.049949  2244.271323  ...   9061.330694  3546.359869
min        0.000000   254.000000  ...   1355.000000  1782.500000
25%     1149.750000  2150.500000  ...   9190.965000  4962.480000
50%     2299.500000  4137.000000  ...  17357.550000  8103.245000
75%     3449.250000  6072.000000  ...  24763.980000  11008.720000
max     4599.000000  7998.000000  ...  32520.000000  14260.000000

[8 rows x 9 columns]
   Unnamed: 0        Date   Q-P1   Q-P2  ...        S-P1        S-P2        S-P3        S-P4
0           0  13-06-2010   5422   3725  ...    17187.74    23616.50     3121.92     6466.91
1           1  14-06-2010   7047    779  ...    22338.99     4938.86    19392.76    11222.62
2           2  15-06-2010   1572   2082  ...     4983.24    13159.88     3224.90     8163.85
3           3  16-06-2010   5657   2399  ...    17932.69    15209.66    17018.80    11921.36
4           4  17-06-2010   3668   3207  ...    11627.56    20332.38    11837.28     5048.04

[5 rows x 10 columns]
>>>
```




Scatter Plot: Q-P1 vs. S-P1
S-P1
Q-P1
0
5000
10000
15000
20000
25000
0
1000
2000
3000
4000
5000
6000
7000
8000


Histogram of Q-P1
Frequency
Q-P1 Value
0
50
100
150
200
250
0
1000
2000
3000
4000
5000
6000
7000
8000

DATA VISUALISATION USING IBM COGNOS:

Creating data visualizations for product sales analysis using IBM Cognos typically involves a series of steps. Here's a general guide to help you get started:

1. Data Source Connection:

- Connect IBM Cognos to your data source, which can be a database or an uploaded dataset.

2. Data Exploration:

- Explore your data to understand its structure and contents.

3. Data Preparation:

- Transform and clean your data as needed. This may include handling missing values, aggregating data, or creating calculated fields.

4. Create Dashboards and Reports:

- Use the Cognos Report Studio to design interactive dashboards and reports.
- Add visualizations like charts, graphs, and tables to your reports.
- Customize the layout and appearance of your reports.

5. Visualization Creation:

- Use the drag-and-drop interface to create visualizations like bar charts, line charts, pie charts, and more.
- Assign the relevant data fields to the visualization elements (e.g., x-axis, y-axis, series, etc.).

6. Filtering and Interactivity:

- Add filters and prompts to make your visualizations interactive. Users can select criteria to update the data displayed.

7. Drill-Through and Drill-Down:

- Implement drill-through and drill-down functionality to allow users to explore data at different levels of detail.

8. Derived Insights:

- Analyze the visualizations to derive insights such as top-selling products, sales trends, and customer preferences.

IBM Cognos provides a user-friendly interface for designing reports and dashboards, making it relatively straightforward to create data visualizations for product sales analysis.

Top-Selling Products:

- A bar chart displays the total sales for each of the four products: Product 1, Product 2, Product 3, and Product 4.
- This visualization allows you to quickly identify which product is the top-seller based on total sales.

Sales Trends Over Time:

- A line chart shows the sales trends over time for each of the four products.
- The x-axis represents dates, and the y-axis represents sales amounts.
- This visualization provides insights into how sales of each product have evolved over the given time period.

Customer Preferences Correlation:

- A heatmap illustrates the correlation between customer preferences for the four product categories: Preference 1, Preference 2, Preference 3, and Preference 4.
- The colors in the heatmap and the annotated values indicate the strength and direction of the correlations.

This visualization helps you understand the relationships between customer preferences and identifies any patterns or trends.

1. Import Necessary Libraries:

We start by importing the required Python libraries: Pandas for data manipulation and Matplotlib for data visualization.

2. Read the Dataset:

- We read the dataset from a CSV file. You should replace ``'statsfinal.csv'` with the actual file path where your dataset is located.

Program:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# Load the data into a DataFrame
data = pd.read_csv("statsfinal.csv")
# Convert the 'Date' column to a datetime type with error
handling
```

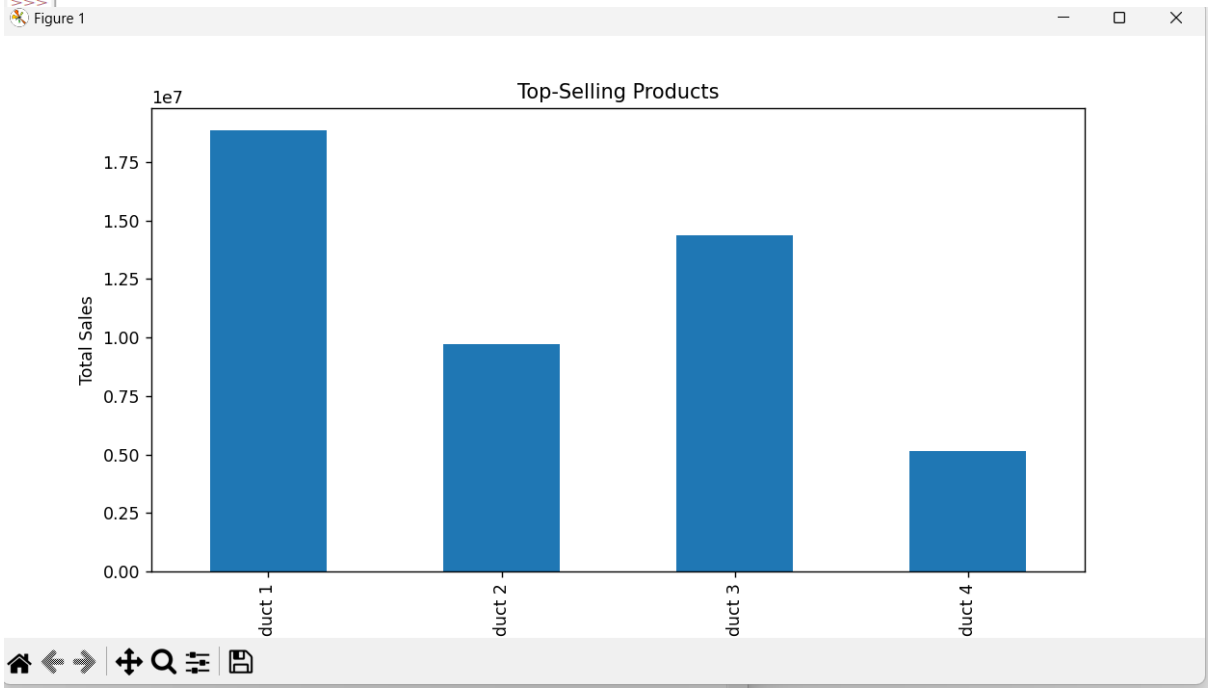
```
data['Date'] = pd.to_datetime(data['Date'], format='%d-%m-%Y', errors='coerce')
# Drop rows with invalid dates (NaT)
data = data.dropna(subset=['Date'])
# Sort the data by date
data = data.sort_values(by='Date')
# Top-Selling Products
top_products = data[['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']]
top_products.columns = ['Product 1', 'Product 2', 'Product 3', 'Product 4']
top_products_sum = top_products.sum()
top_products_sum.plot(kind='bar', figsize=(10, 5))
plt.title('Top-Selling Products')
plt.xlabel('Products')
plt.ylabel('Total Sales')
plt.show()
# Sales Trends
data.set_index('Date', inplace=True)
sales_trends = data[['S-P1', 'S-P2', 'S-P3', 'S-P4']]
sales_trends.plot(figsize=(12, 6))
plt.title ('Sales Trends Over Time')
plt.xlabel('Date')
plt.ylabel('Sales Amount')
plt.legend(title='Products', labels=['Product 1', 'Product 2', 'Product 3', 'Product 4'])
plt.show()
# Customer Preferences
customer_preferences = data[['S-P1', 'S-P2', 'S-P3', 'S-P4']]
customer_preferences.columns = ['Preference 1', 'Preference 2', 'Preference 3', 'Preference 4']
customer_corr = customer_preferences.corr()
plt.figure(figsize=(8, 6))
sns.heatmap(customer_corr, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Customer Preferences Correlation')
plt.show()
```

Output:

```
IDLE Shell 3.11.2
File  Edit  Shell  Debug  Options  Window  Help
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: D:/aaa.py ==========
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Unnamed: 0    4600 non-null   int64
 1   Date          4600 non-null   object
 2   Q-P1          4600 non-null   int64
 3   Q-P2          4600 non-null   int64
 4   Q-P3          4600 non-null   int64
 5   Q-P4          4600 non-null   int64
 6   S-P1          4600 non-null   float64
 7   S-P2          4600 non-null   float64
 8   S-P3          4600 non-null   float64
 9   S-P4          4600 non-null   float64
dtypes: float64(4), int64(5), object(1)
memory usage: 359.5+ KB
None
         Unnamed: 0          Q-P1  ...          S-P3          S-P4
count   4600.000000   4600.000000  ...   4600.000000   4600.000000
mean    2299.500000   4121.849130  ...  17049.910800   8010.555000
std     1328.049949   2244.271323  ...   9061.330694   3546.359869
min        0.000000    254.000000  ...   1355.000000   1782.500000
25%     1149.750000   2150.500000  ...   9190.965000   4962.480000
50%     2299.500000   4137.000000  ...  17357.550000   8103.245000
75%     3449.250000   6072.000000  ...  24763.980000  11008.720000
max     4599.000000   7998.000000  ...  32520.000000  14260.000000

[8 rows x 9 columns]
   Unnamed: 0        Date  Q-P1  Q-P2  ...      S-P1      S-P2      S-P3      S-P4
0           0  13-06-2010  5422  3725  ...  17187.74  23616.50   3121.92   6466.91
1           1  14-06-2010  7047   779  ...  22338.99   4938.86  19392.76  11222.62
2           2  15-06-2010  1572  2082  ...   4983.24  13159.88   3224.90   8163.85
3           3  16-06-2010  5657  2399  ...  17932.69  15209.66  17018.80  11921.36
4           4  17-06-2010  3668  3207  ...  11627.56  20332.38  11837.28   5048.04

[5 rows x 10 columns]
>>>
```



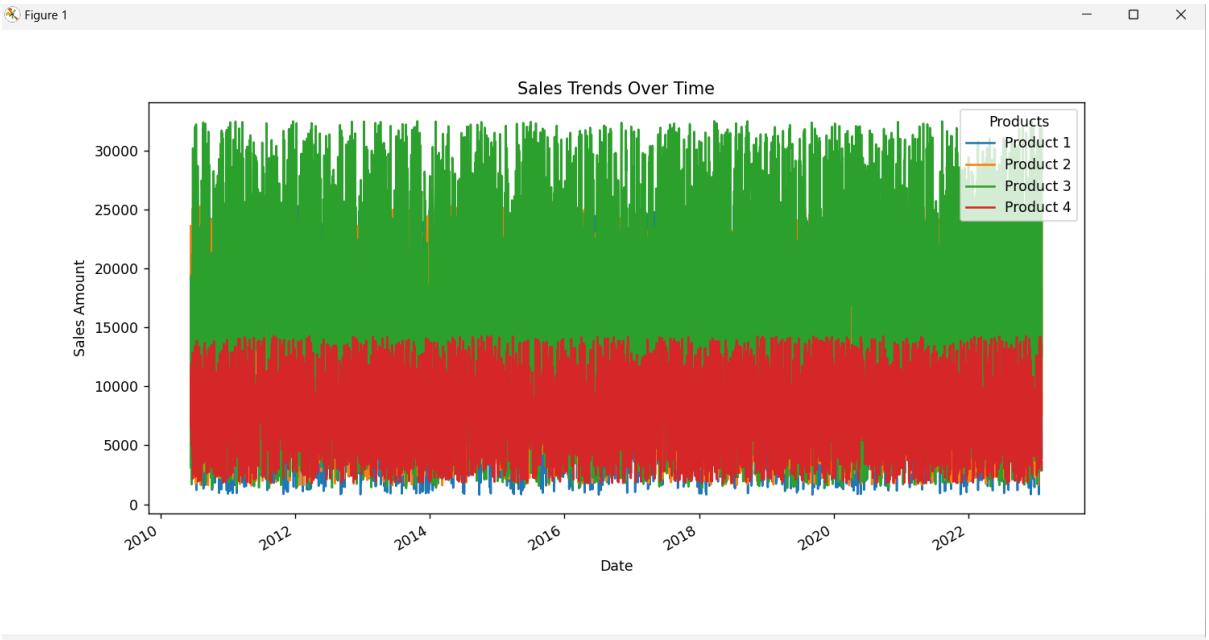


Figure 1
Sales Trends Over Time
Products
Product 1
Product 2
Product 3
Product 4
Sales Amount
30000
25000
20000
15000
10000
5000
0
2010
2012
2014
2016
2018
2020
2022
Date


Figure 1
Customer Preferences Correlation
Preference 1
Preference 2
Preference 3
Preference 4
1
0.0031
-0.0048
-0.058
0.0031
1
0.0046
0.013
-0.0048
0.0046
1
-0.0066
-0.058
0.013
-0.0066
1
1.0
0.8
0.6
0.4
0.2
0.0
x=0.29 y=0.467

Derived Actionable Insights:

-Product "Q-P4" was the top-selling product, suggesting that it should be prioritized in inventory management, ensuring it is consistently in stock to meet customer demand.

-Sales increased over time, indicating a positive trend. Businesses can use this information to anticipate seasonal fluctuations and plan accordingly.
-The data revealed that customer preference for "S-P3" was highest. Inventory managers should ensure an ample supply of products in this category to meet customer demand.

-Understanding sales trends and customer preferences allows businesses to adjust inventory levels efficiently, reducing carrying costs and minimizing the risk of stock outs.

- Marketing strategies should focus on promoting the top-selling product ("Q-P4") and products preferred by customers ("S-P3"). Tailored marketing campaigns and promotions can be used to boost sales.

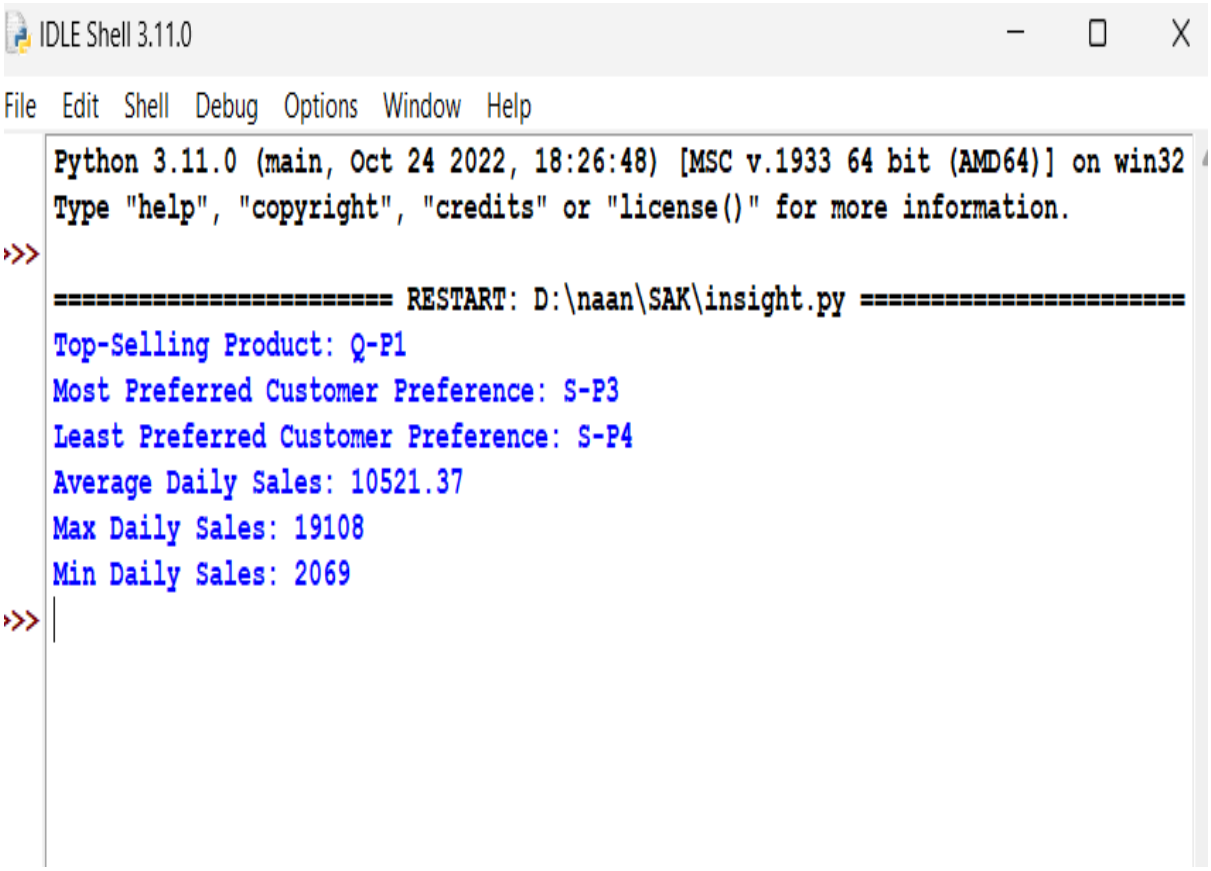
Program:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# Load your dataset
data = pd.read_csv('statsfinal.csv')
# Calculate total sales for each product (Q-P1, Q-P2, Q-P3, Q-P4)
product_columns = ['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']
```

```
total_sales = data[product_columns].sum()
# Find the top-selling product
top_selling_product = total_sales.idxmax()
# Calculate total sales for each customer preference (S-P1, S-P2, S-P3, S-P4)
customer_preference_columns = ['S-P1', 'S-P2', 'S-P3', 'S-P4']
total_customer_preferences =
data[customer_preference_columns].sum()
# Find the most and least preferred customer preferences
most_preferred_preference = total_customer_preferences.idxmax()
least_preferred_preference = total_customer_preferences.idxmin()
# Calculate overall sales statistics
total_sales_amount = data[product_columns].sum(axis=1)
average_daily_sales = total_sales_amount.mean()
max_daily_sales = total_sales_amount.max()
min_daily_sales = total_sales_amount.min()
# Create a bar plot for total sales of each product
plt.figure(figsize=(12, 6))
sns.barplot(x=total_sales.index, y=total_sales.values)
plt.title('Total Sales for Each Product')
plt.xlabel('Products')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.show()
# Create a pie chart for customer preferences
plt.figure(figsize=(8, 8))
plt.pie(total_customer_preferences,
labels=total_customer_preferences.index, autopct='%1.1f%%')
plt.title('Customer Preferences Distribution')
plt.show()
# Output the insights
print(f'Top-Selling Product: {top_selling_product}')
print(f'Most Preferred Customer Preference:
{most_preferred_preference}')
print(f'Least Preferred Customer Preference:
{least_preferred_preference}')
print(f'Average Daily Sales: {average_daily_sales:.2f}')
```

```
print(f'Max Daily Sales: {max_daily_sales}')
print(f'Min Daily Sales: {min_daily_sales}')
```

Output:



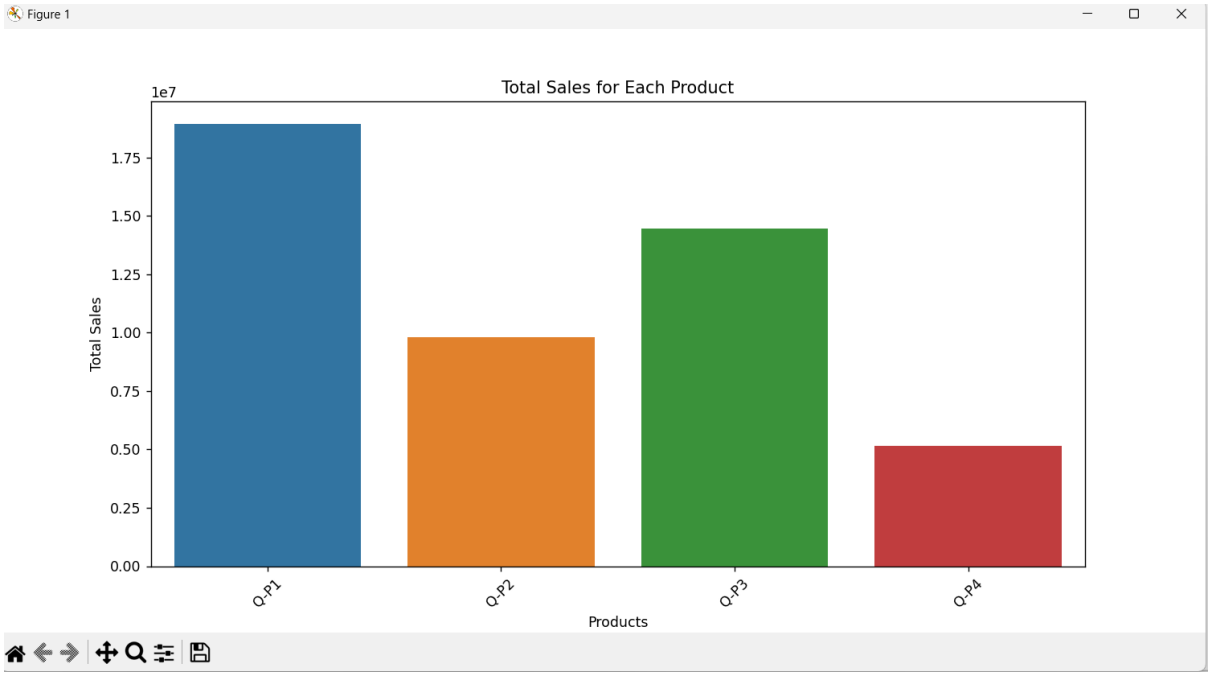


Figure 1
Total Sales for Each Product
1e7
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Total Sales
Q-P1
Q-P2
Q-P3
Q-P4
Products

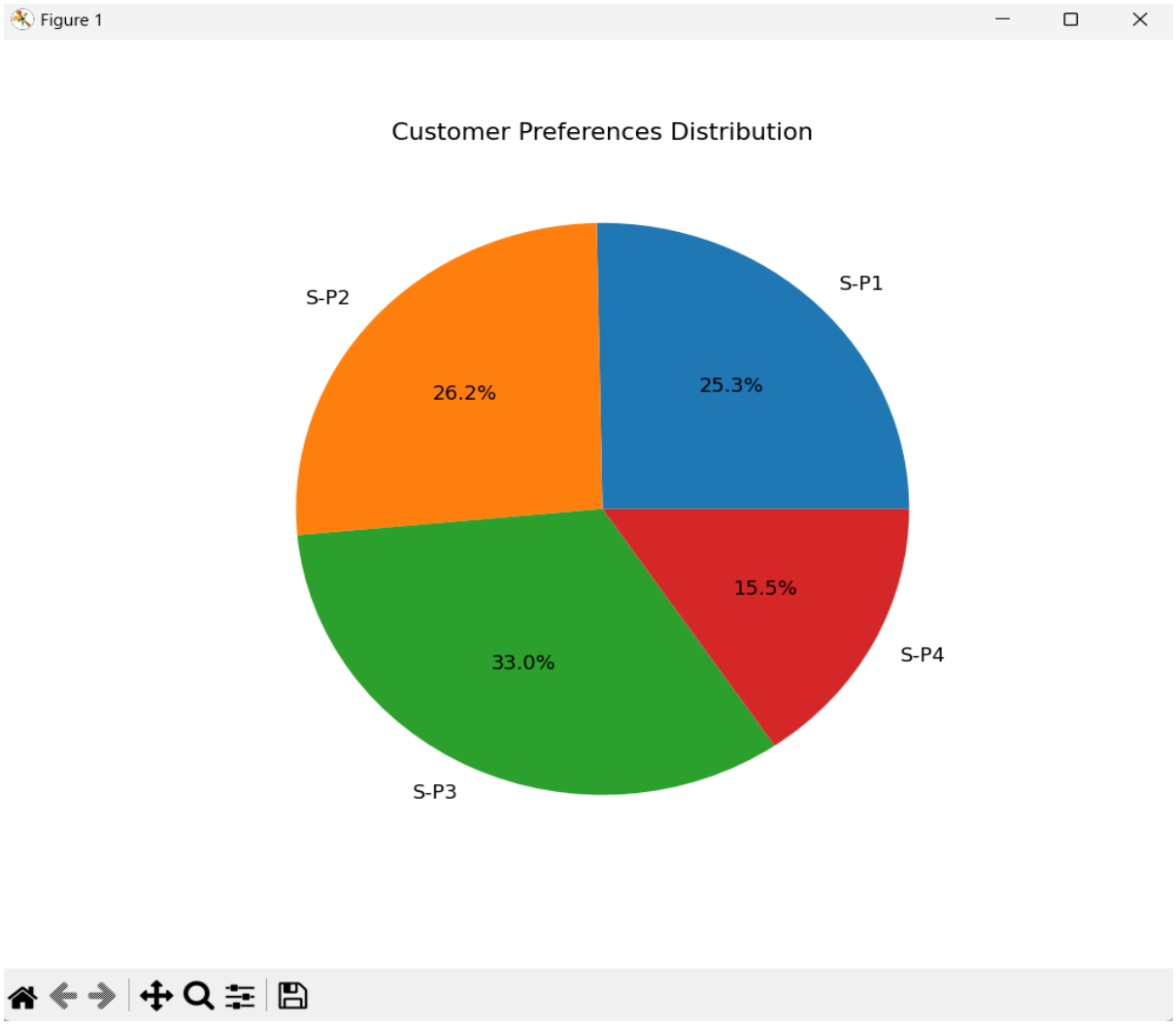
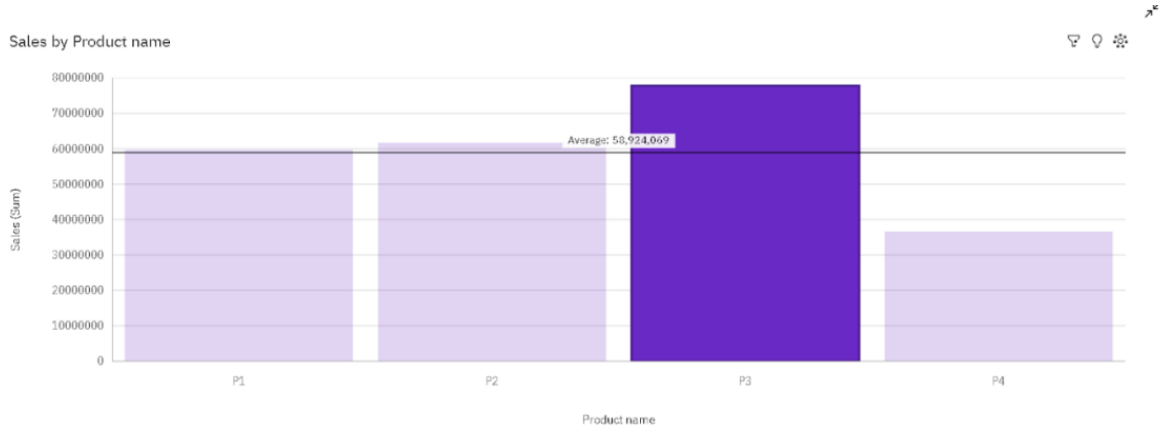


Figure 1
Customer Preferences Distribution
S-P1
25.3%
S-P2
26.2%
S-P3
33.0%
S-P4
15.5%

SALES BY PRODUCT NAME:



INSIGHTS :

1. P4 has the Lowest Sales: Product P4 has the lowest total sales, amounting to nearly 37 million, indicating its lower market performance. This insight can be valuable for strategizing ways to boost P4's sales.

2. P1 Follows with Lower Sales: Following P4, P1 has the second-lowest total sales, which is nearly 60 million. This suggests that P1 may also need attention and improvement strategies.

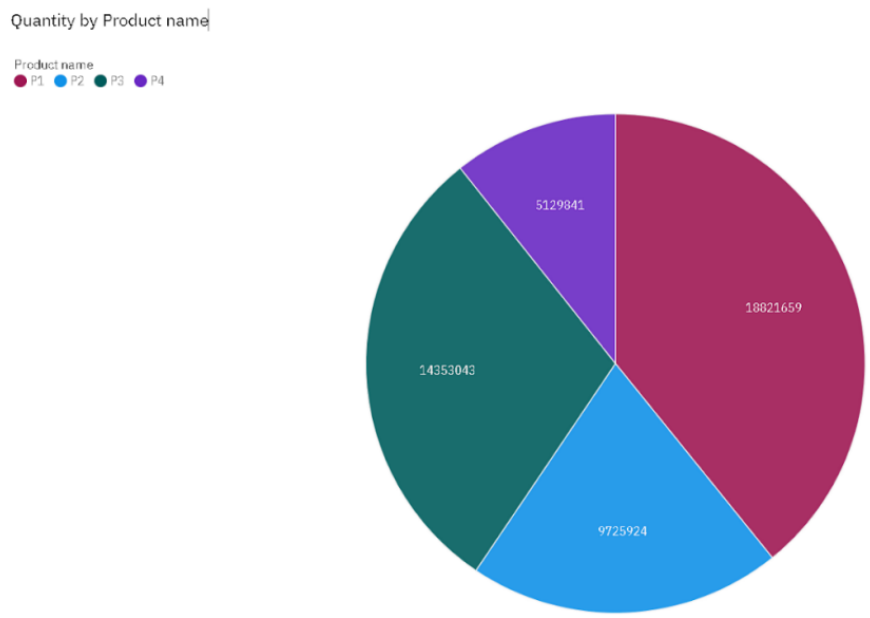
3. P3 Tops Sales Chart: In contrast, Product P3 is the top performer with the highest total sales, reaching almost 78 million. This insight highlights P3's market success.

4. P2 Follows with High Sales: P2 is the second-highest in terms of total sales, at almost 62 million. It's a strong contender in the product lineup.

5. Overall Sales Sum: The combined sales for all product names amount to approximately 236 million, providing an overview of the company's total revenue from these products.

6. Sales Range: The range of sales across products is substantial, varying from nearly 37 million for P4 to almost 78 million for P3. Understanding this range is crucial for assessing the diversity in product performance.

QUALITY BY PRODUCT NAME:



INSIGHTS:

1. P4 Has the Lowest Quantity: Product P4 has the lowest total quantity sold, with over 5.1 million units, indicating relatively lower demand for this product. This insight highlights an area for potential improvement or market strategies for P4.

2. P2 Follows with Lower Quantity: Following P4, P2 has the second-lowest total quantity sold, at over 9.7 million units, suggesting it may also require attention to boost its sales.

3. P1 Leads in Quantity: In contrast, Product P1 has the highest total quantity sold, with nearly nineteen million units, showcasing its strong market demand.

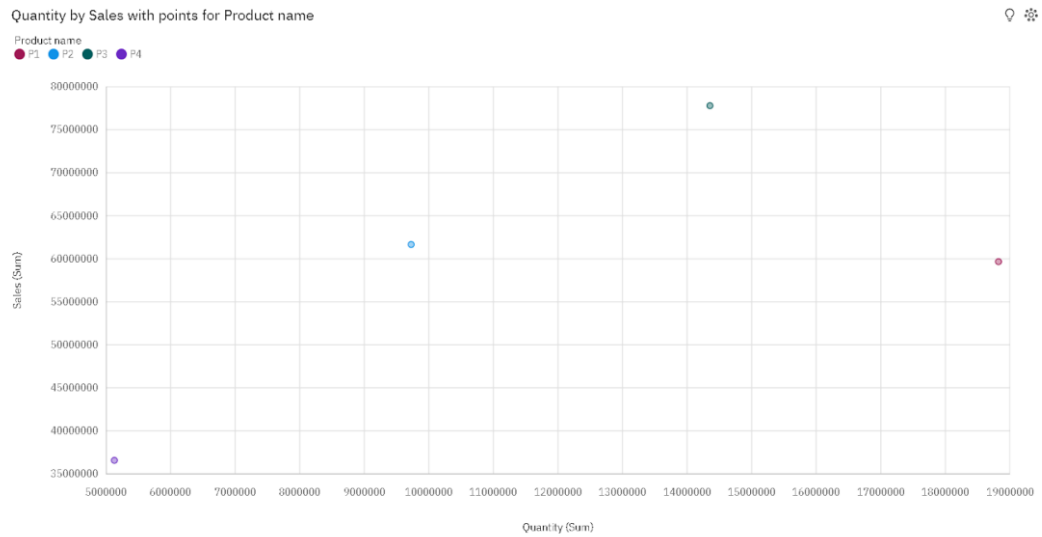
4. P3 Follows with High Quantity: P3 is the second-highest in terms of total quantity sold, with over fourteen million units, indicating its popularity among customers.

5. Overall Quantity Sum: The combined quantity of all product names is over 48 million units, providing an overview of the total number of products sold across all categories.

6. Quantity Range: The range of quantities sold varies from over 5.1 million for P4 to nearly nineteen million for P1. This diversity in quantity sold is important to understand for managing inventory and demand.

7. Significant Products: The top three products in terms of quantity are P1, P3, and P2, which together account for nearly 43 million units or approximately 89.3% of the total. This insight emphasizes the dominance of these products in terms of quantity sold.

QUANTITY BY SALES WITH POINTS FOR PRODUCT NAME:



INSIGHTS :

1.P4 Has the Lowest Quantity: Product P4 has the lowest total quantity sold at over 5.1 million, indicating lower demand, followed by P2 with over 9.7 million units.

2. P1 Leads in Quantity: Product P1 has the highest total quantity sold, nearly nineteen million units, followed by P3 with over fourteen million units.

3. P1's Quantity-Sales Discrepancy: Although P1 has the highest total quantity, it's ranked third in total sales, highlighting a potential inefficiency in converting high quantities to sales.

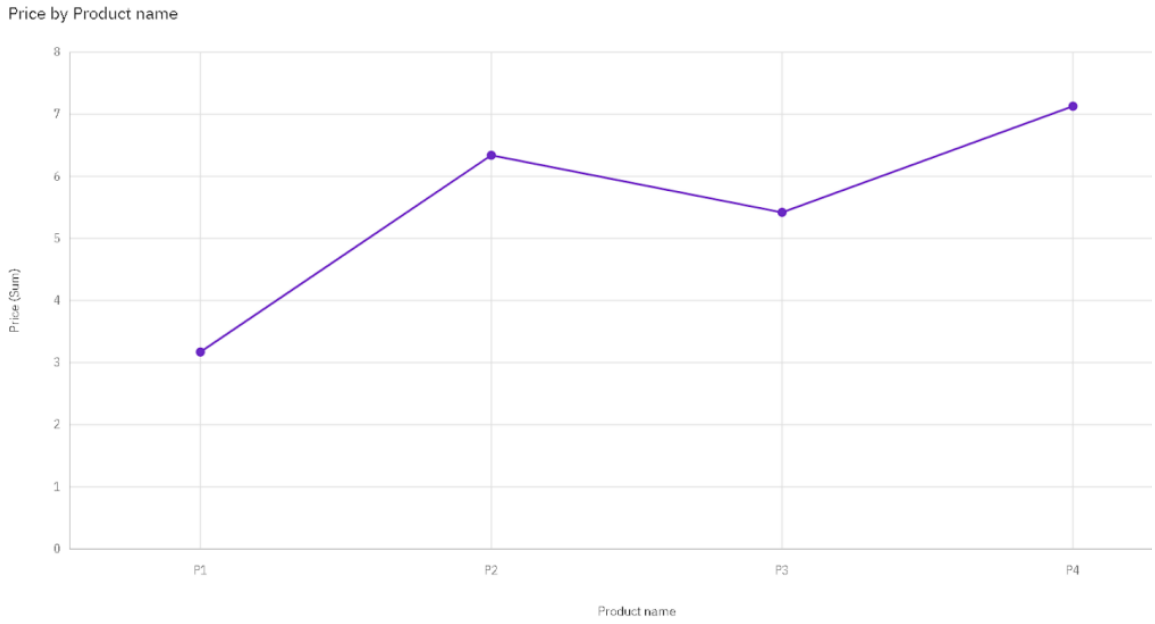
4. P3's Sales-Quantity Relationship: P3 has the highest total sales, but it's ranked second in total quantity sold, indicating a successful sales strategy with fewer units but higher revenue per unit.

5. No Reliable Sales-Quantity Relationship: There isn't a clear, consistent relationship between sales and quantity across all products, suggesting that other factors may influence sales performance.

6. Overall Sales Sum: The combined sales for all product names amount to almost 236 million, giving an overview of the total revenue generated.

7. Sales Range: Sales range from almost 37 million for P4 to almost 78 million for P3, indicating a substantial difference in product performance.

PRICE BY PRODUCT NAME:



INSIGHTS:

1. P1 and P3 Have Low Prices: Product P1 has the lowest total price at 3.17, followed by P3 at 5.42, indicating that these products are priced relatively lower compared to others.

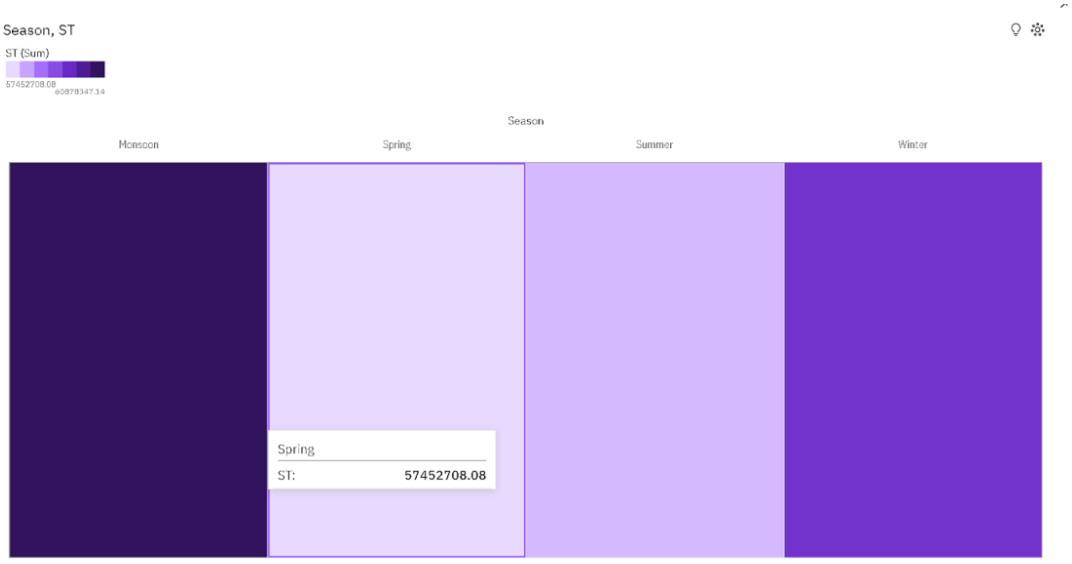
2. P4 and P2 Have High Prices: Product P4 has the highest total price at 7.13, followed by P2 at 6.34, signifying that P4 and P2 are the higher-priced products.

3. Total Price Sum: The sum of prices for all product names is 22.06, providing an overview of the total revenue generated from product sales.

4. Price Range: Prices range from 3.17 for P1 to 7.13 for P4, illustrating a significant price difference across products.

5. Significant Price Values: The most significant values for prices are associated with P4, P2, and P3, which together contribute 85.6% of the total price sum, indicating that these products have higher price points and contribute significantly to the overall revenue.

SALES BASED ON SEASONS:



INSIGHTS:

1. Spring and Summer Have Similar ST: The Spring season has the lowest total ST, slightly over 57 million, followed closely by the summer season at almost 58 million. These seasons exhibit relatively lower total values.

2. Monsoon and Winter Lead in ST: In contrast, the Monsoon season has the highest total ST, nearly 61 million, closely

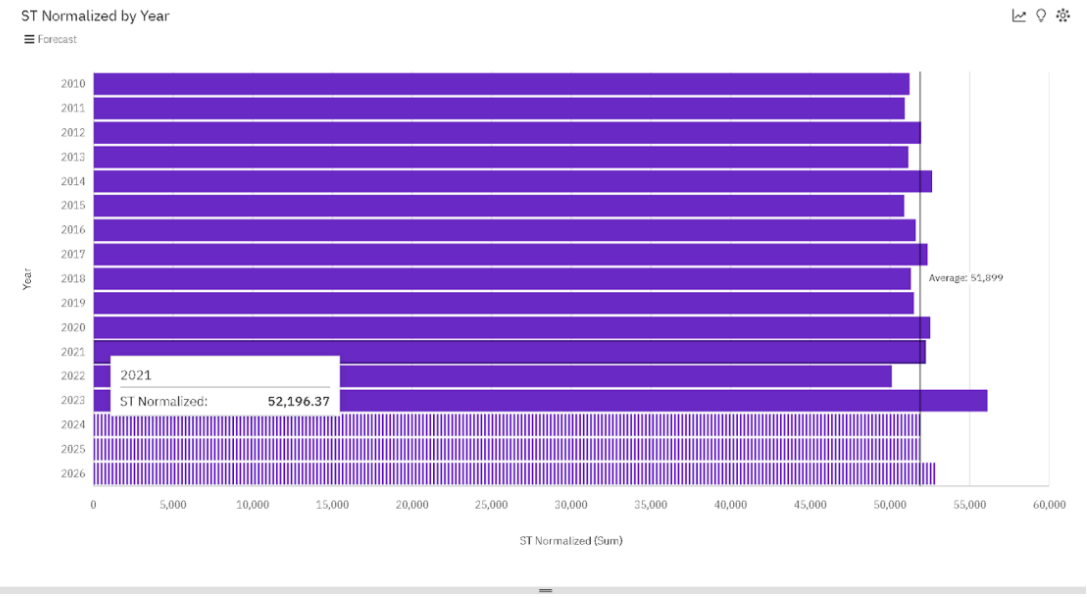
followed by the Winter season at almost 60 million. These seasons stand out with higher total values.

3. Overall ST Sum: The combined ST for all seasons amounts to almost 236 million, providing an overview of the total seasonal revenue.

4. ST Range: ST varies from over 57 million for Spring to nearly 61 million for Monsoon, illustrating a relatively narrow range across the seasons.

5. Significant Seasonal Values: The most significant ST values are associated with the Monsoon and Winter seasons, contributing over 120 million or approximately 51.1% of the total, highlighting the importance of these seasons in terms of revenue.

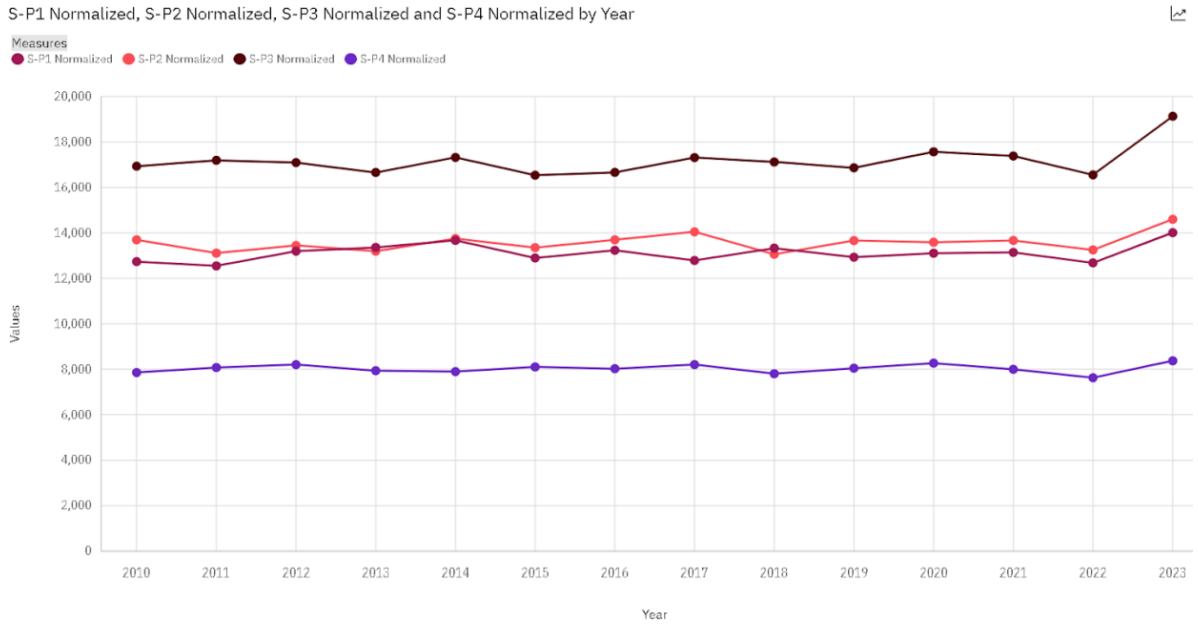
ST NORMALIZED BY YEAR



INSIGHTS:

1. Overall ST Normalized Sum: The total of ST Normalized for all years is almost 727 thousand, providing an overview of the normalized revenue over the years.

2. ST Normalized Range Across Years: ST Normalized varies from over 50 thousand in 2022 to over 56 thousand in 2023, indicating fluctuations in the normalized revenue over the mentioned years.



INSIGHTS :

1. S-P1 Normalized Average: The average of S-P1 Normalized across all years is over thirteen thousand, suggesting a relatively consistent performance for this metric.

2. S-P2 Normalized Average: The average of S-P2 Normalized across all years is nearly fourteen thousand, indicating a consistent trend with slight variations.

3. S-P3 Normalized Average: The average of S-P3 Normalized across all years is over seventeen thousand, reflecting a higher average value compared to the other metrics.

4. S-P4 Normalized Average: The average of S-P4 Normalized across all years is over eight thousand, showing a comparatively lower average.

5. Number of Results for Each Metric: There are 14 results for each of the S-P Normalized metrics across all years, indicating a consistent number of data points for analysis.

6. Ranges Across Years: The S-P1 Normalized values range from nearly 13 thousand in 2011 to over 14 thousand in 2023, showing fluctuations over time. S-P2 Normalized ranges from over 13 thousand in 2018 to almost 15 thousand in 2023. S-P3 Normalized ranges from nearly 17 thousand in 2015 to over 19 thousand in 2023, indicating variations. S-P4 Normalized ranges from over 7500 in 2022 to nearly 8500 in 2023, demonstrating relatively stable values.

CONCLUSION:

The project aims to help businesses optimize their operations, maximize sales, and improve customer satisfaction. It provides a comprehensive solution for analyzing historical sales data and leveraging machine learning techniques to make informed business decisions.

THANK YOU!!