PRODUCT SALES ANALYSIS

ABSTRACT

- The "Product Sales Analysis" machine learning project aims to develop a predictive model that can analyze and forecast product sales based on historical data.

- This project utilizes a dataset containing information about product attributes, sales channels, pricing, and time-related factors.

OBJECTIVES

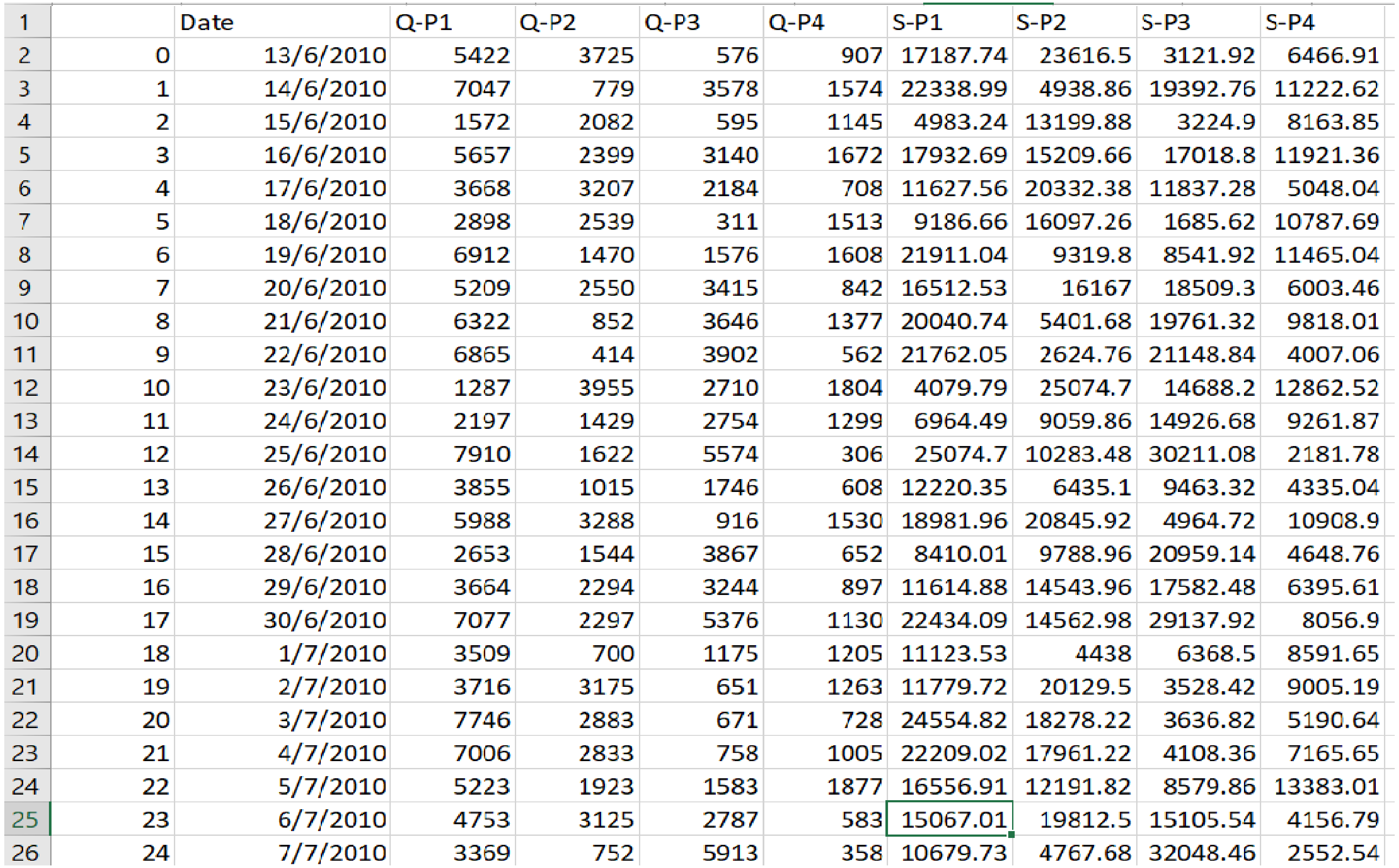
- Product sales analysis typically has several objectives, including:

Performance Evaluation, Identifying Trends, Customer Insights, Inventory Management, Competitive Analysis, Profitability Analysis, Marketing Effectiveness, Forecasting, Geographic Analysis, Product Lifecycle Management, Customer Retention, Identifying Growth Opportunities, Cost Reduction, Quality Improvement, Compliance and Reporting

- By achieving these goals we would know about the sales, profit of the products.

Data Source

Dataset Link:https://www.kaggle.com/datasets/ksabishek/product-sales-data



1		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

DATA PREPROCESSING

- Clean the dataset: Check for missing values and outliers.
- Convert the 'Date' column to a datetime 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.

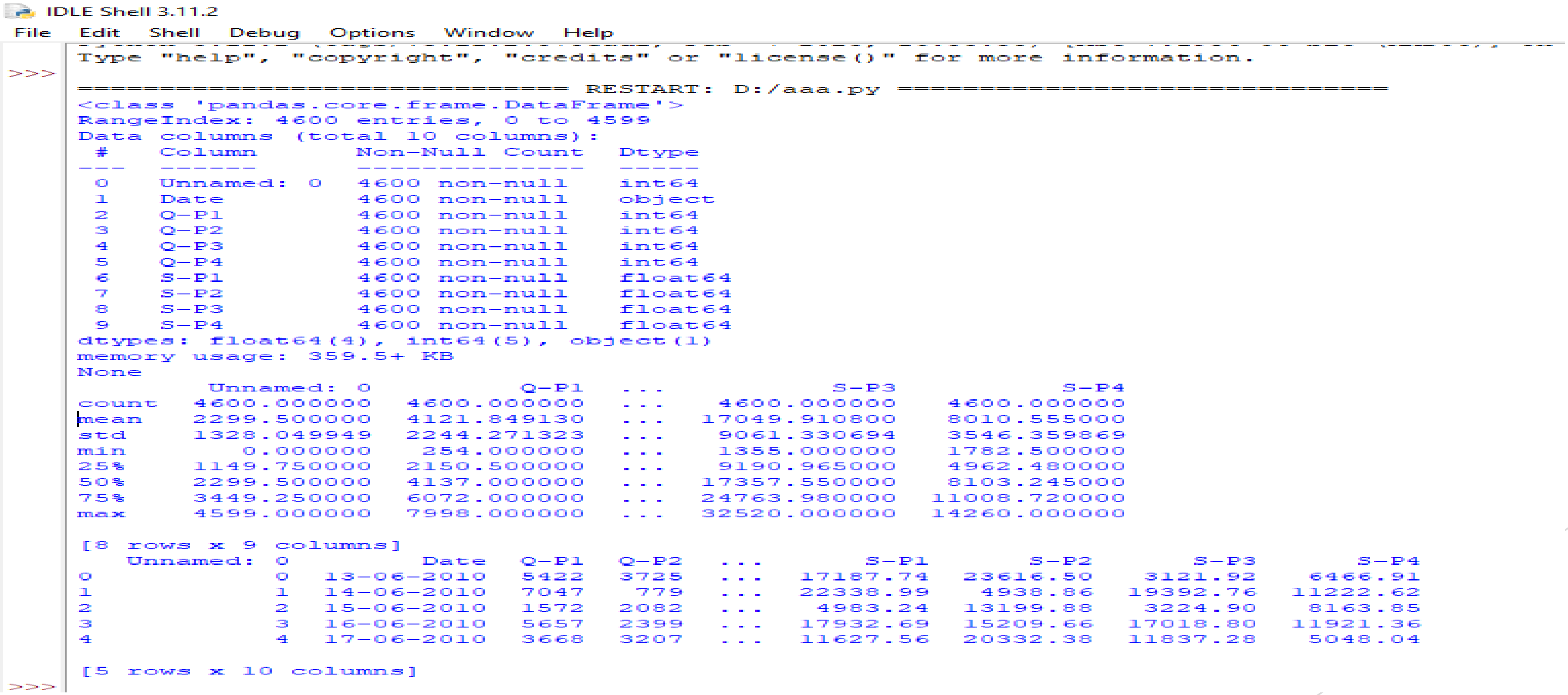
Program:

```
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'])
```

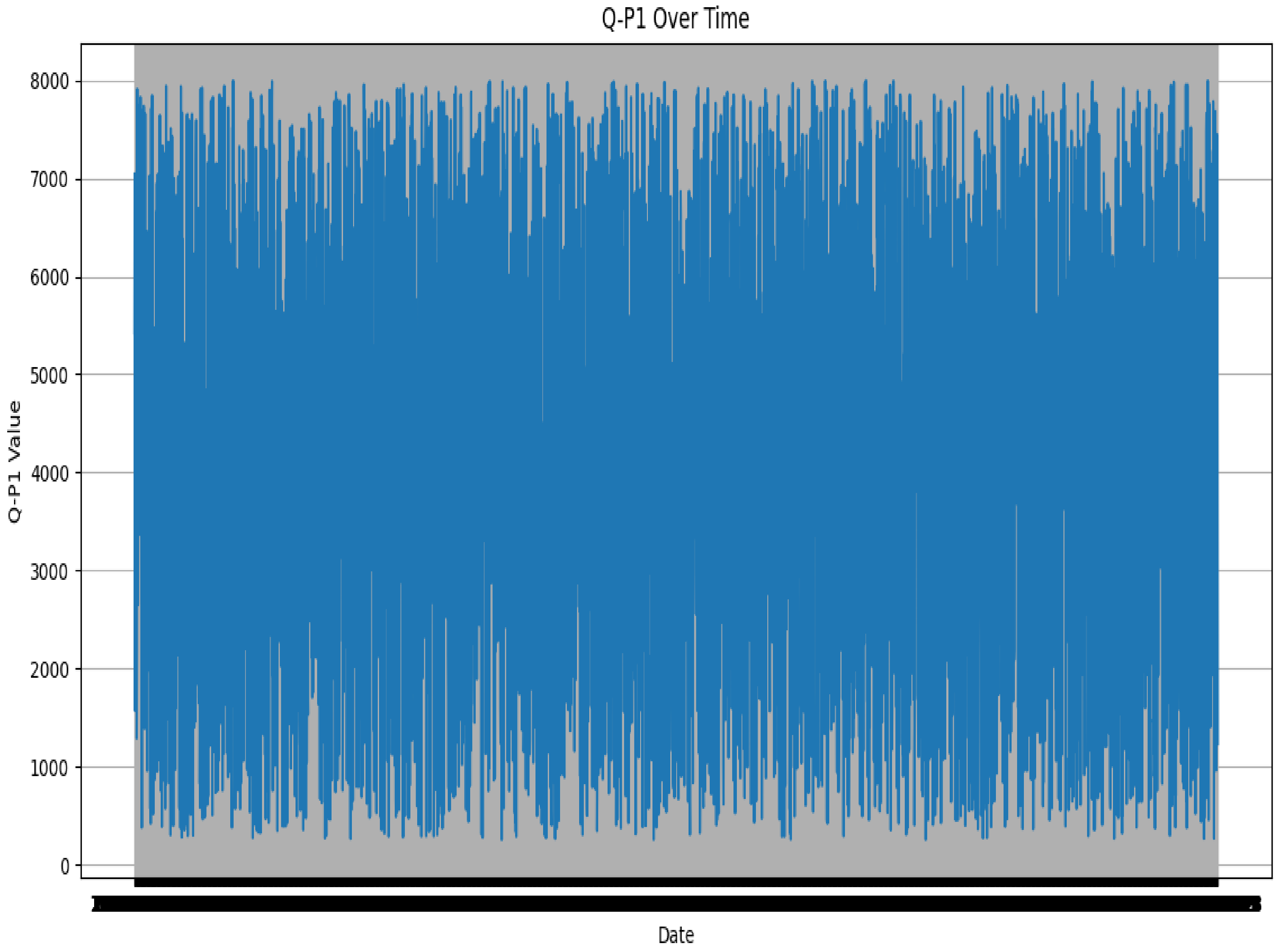
```
print(data.info())
print(data.describe())
print(data.head())

# 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()
```

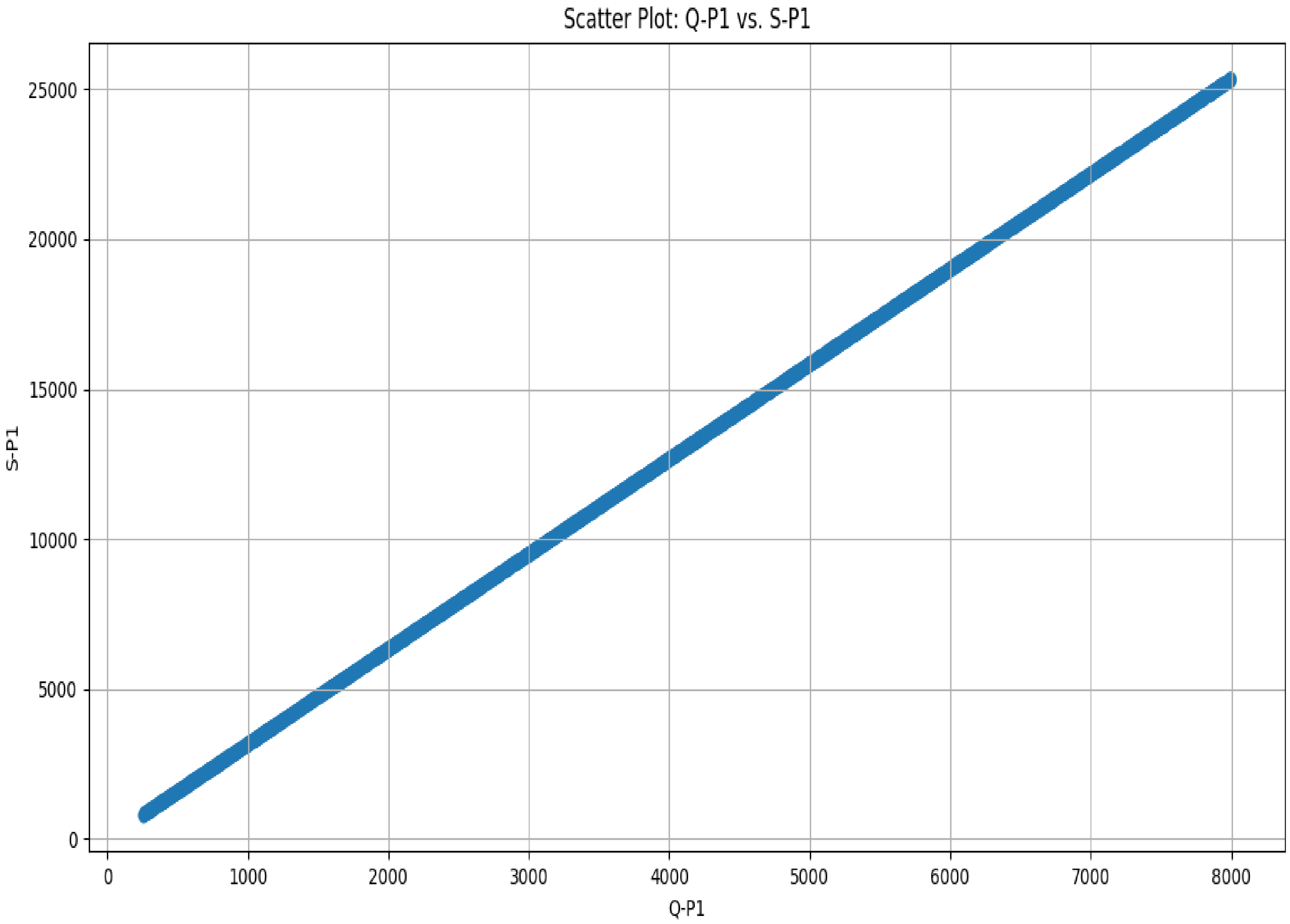
```
# Visualization 3: Histogram of a numeric column (e.g., Q-P1)
plt.figure(figsize=(10, 6))
plt.hist(data['Q-P1'], bins=20, edgecolor='k')
plt.title('Histogram of Q-P1')
plt.xlabel('Q-P1 Value')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

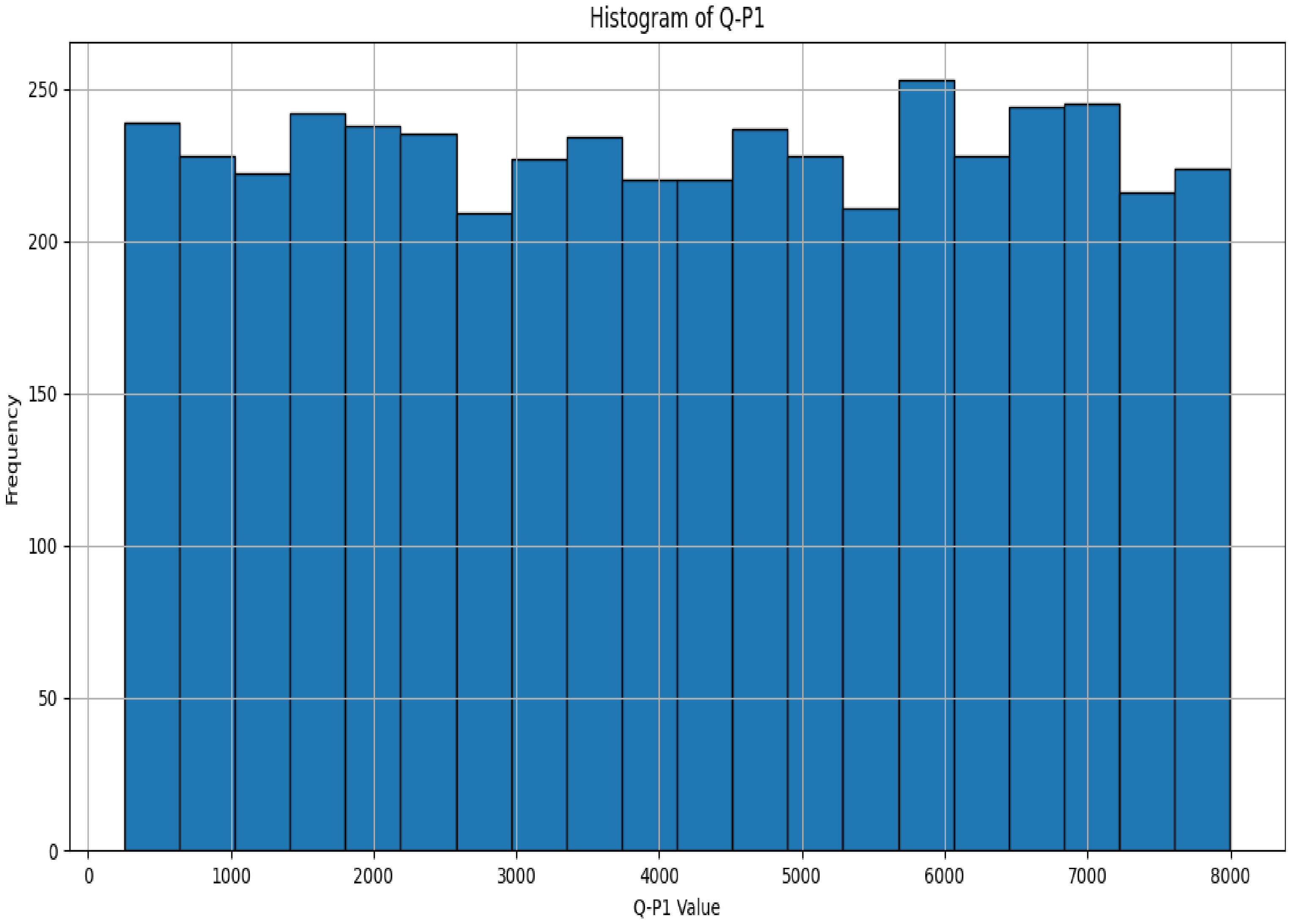
OUTPUT:



```
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  13199.88   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]
>>>
```


Q-P1 Over Time
Q-P1 Value
8000
7000
6000
5000
4000
3000
2000
1000
0
Date

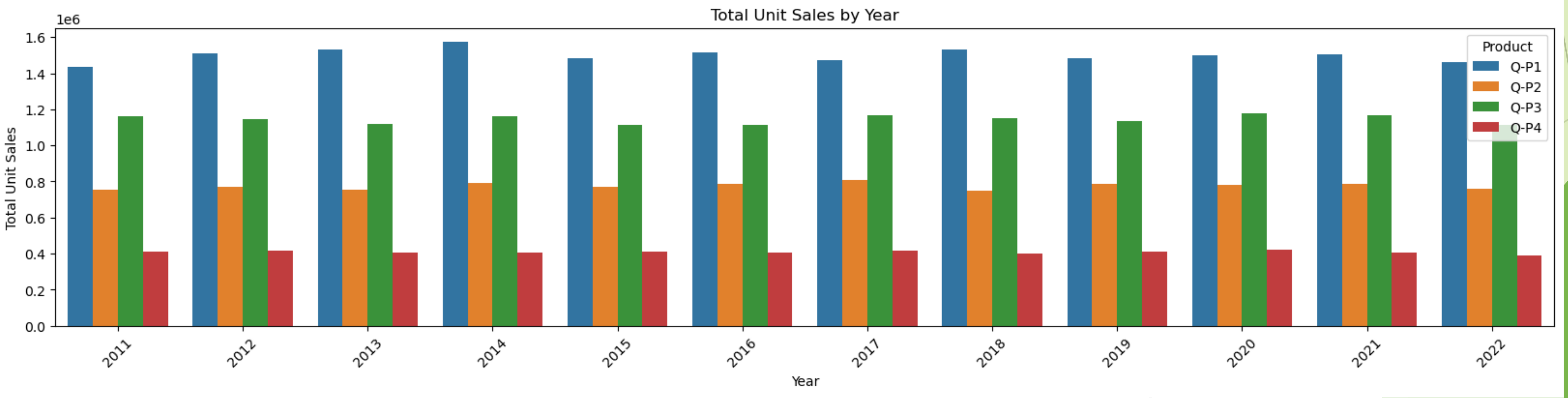

Scatter Plot: Q-P1 vs. S-P1
S-P1
Q-P1
0
5000
10000
15000
20000
25000
0
1000
2000
3000
4000
5000
6000
7000
8000

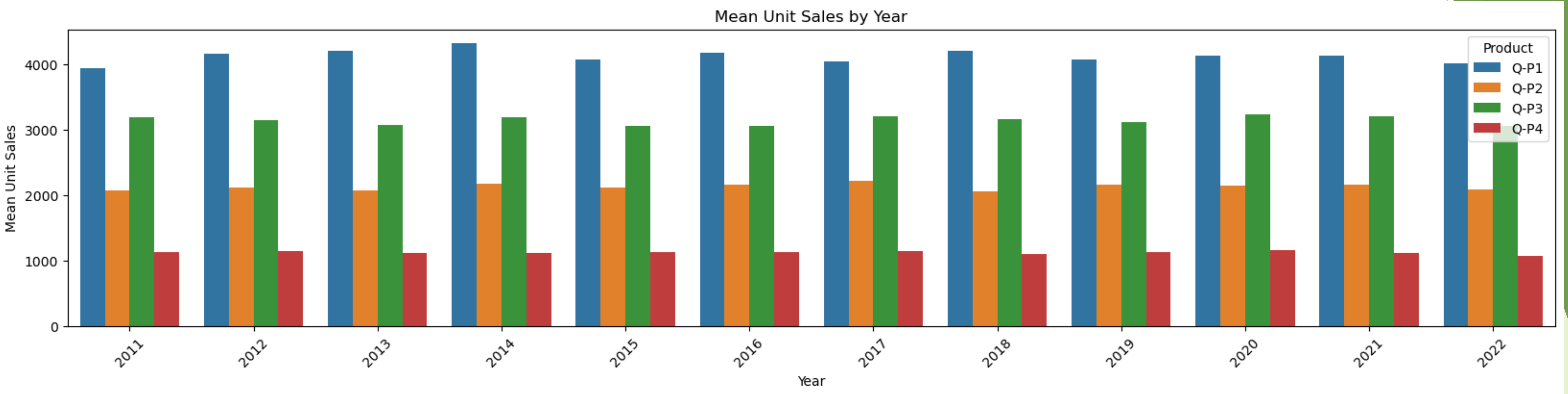

Histogram of Q-P1
Frequency
0
50
100
150
200
250
0
1000
2000
3000
4000
5000
6000
7000
8000
Q-P1 Value

EXPLORATORY DATA ANALYSIS (EDA)

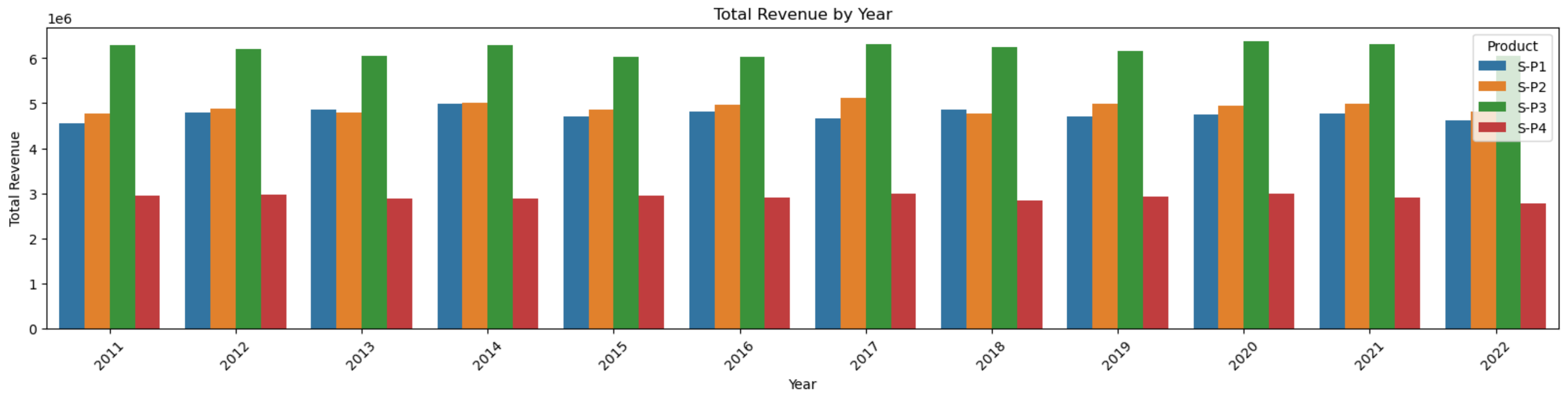
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
data = pd.read_csv('/kaggle/input/product-sales-data/statsfinal.csv'
data['Day'] = data['Date'].apply(lambda x: x.split('-')[0])
data['Month'] = data['Date'].apply(lambda x: x.split('-')[1])
data['Year'] = data['Date'].apply(lambda x: x.split('-')[2])
data_reduced = data.query("Year != '2010' and Year != '2023'")
def plot_bar_chart(df, columns, stri, str1, val):
if val == 'sum': sales_by_year = df.groupby('Year')[columns].sum().reset_index()
elif val == 'mean': sales_by_year = df.groupby('Year')[columns].mean().reset_index()
```

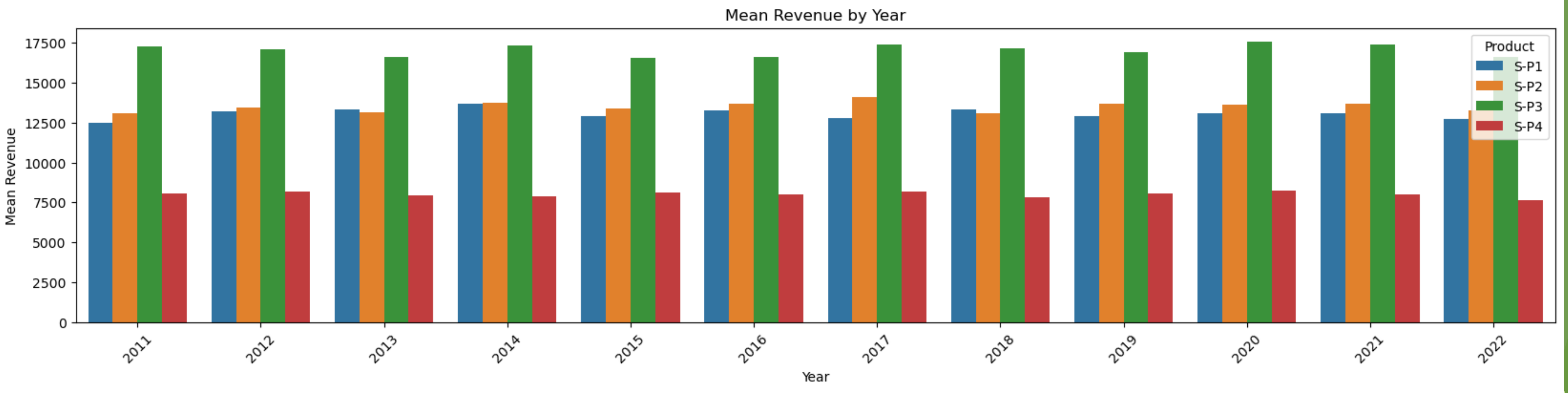
```
sales_by_year_melted = pd.melt(sales_by_year, id_vars='Year', value_vars=columns,
var_name='Product', value_name='Sales')

plt.figure(figsize=(20,4))

sns.barplot(data=sales_by_year_melted, x='Year', y='Sales', hue='Product')

plt.xlabel('Year')

plt.ylabel(stri)

plt.title(f'{stri} by {str1}')

plt.xticks(rotation=45)

plt.show()

plot_bar_chart(data_reduced, ['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4'],'Total Unit Sales', 'Year', 'sum')

plot_bar_chart(data_reduced, ['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4'],'Mean Unit Sales', 'Year', 'mean')
```



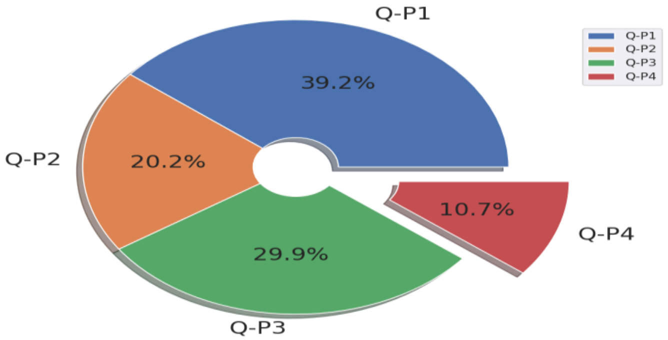


```
plot_bar_chart(data_reduced, ['S-P1', 'S-P2', 'S-P3', 'S-P4'], 'Total Revenue', 'Year', 'sum')
plot_bar_chart(data_reduced, ['S-P1', 'S-P2', 'S-P3', 'S-P4'], 'Mean Revenue', 'Year', 'mean')
```

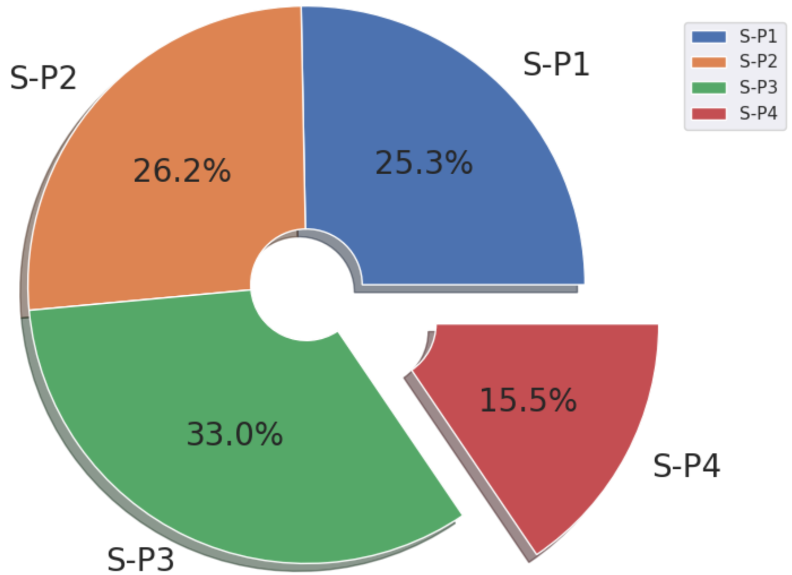




```
q = df[["Q-P1","Q-P2","Q-P3","Q-P4"]].sum()
print(q)
plt.figure(figsize=(8,8))
plt.pie(q,labels=df[["Q-P1","Q-P2","Q-P3","Q-
P4"]].sum().index,shadow=True,autopct="%0.01f%%",textprops={"fontsize":20},wedgeprops={'width':
0.8},explode=[0,0,0,0.3])
plt.legend(loc='center right', bbox_to_anchor=(1.2, 0.8));
```

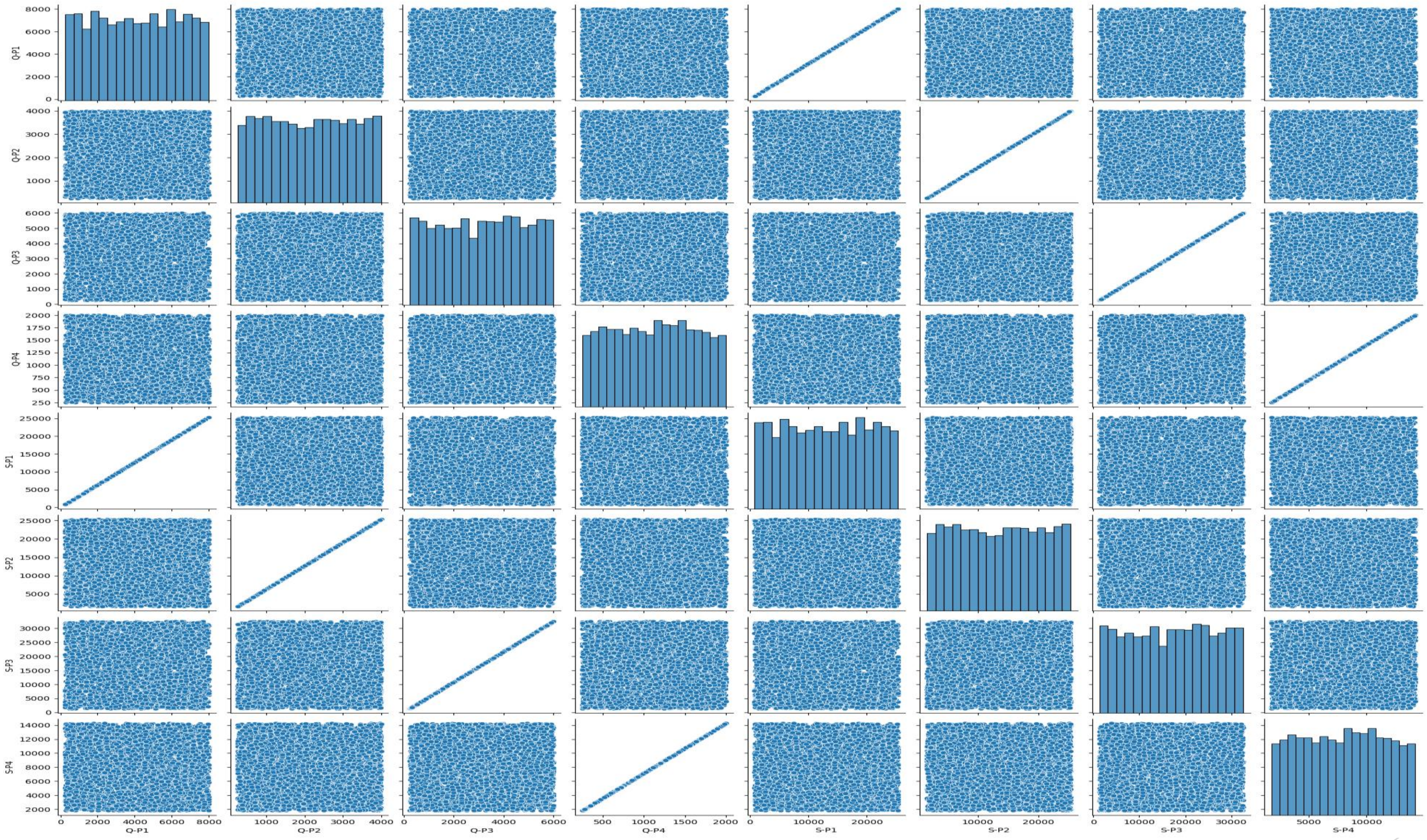


```
s=df[["S-P1","S-P2","S-P3","S-P4"]].sum()
print(s)
plt.figure(figsize=(8,8))
plt.pie(s,labels=df[["S-P1","S-P2","S-P3","S-
P4"]].sum().index,shadow=True,autopct="%0.01f%%",textprops={"fontsize":20},wedgeprops={'width':
0.8},explode=[0,0,0,0.3])
plt.legend(loc='center right', bbox_to_anchor=(1.2, 0.8))
```

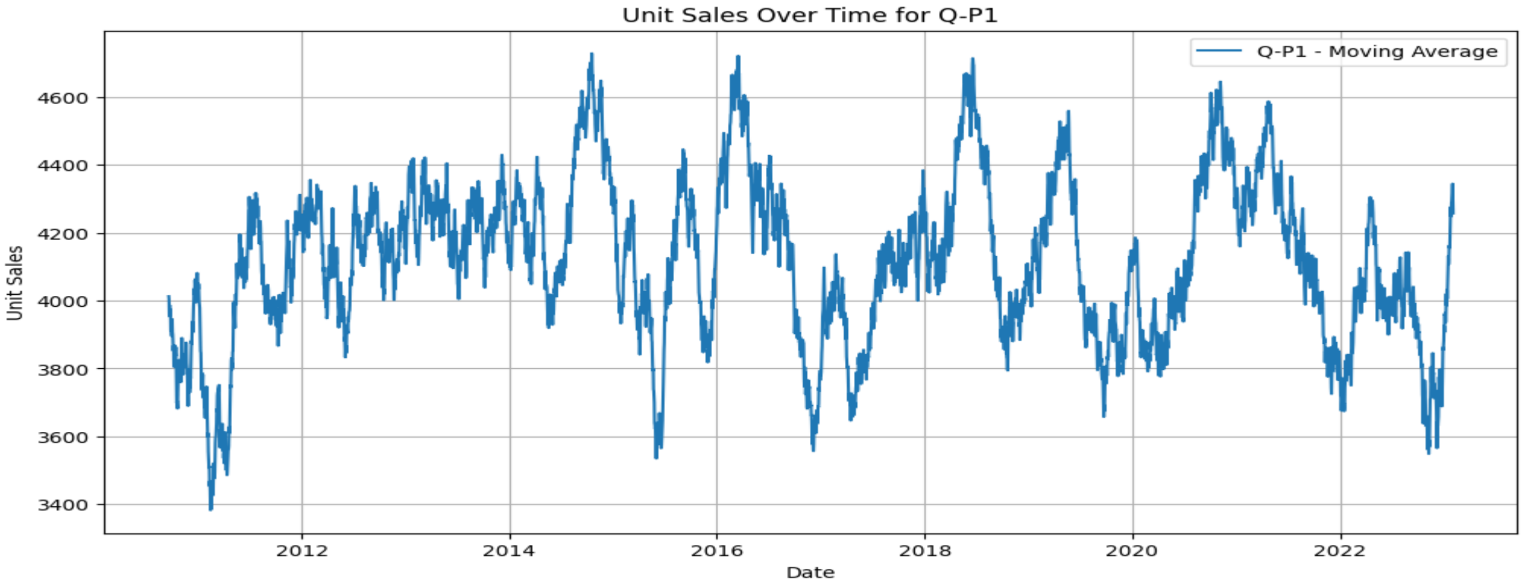


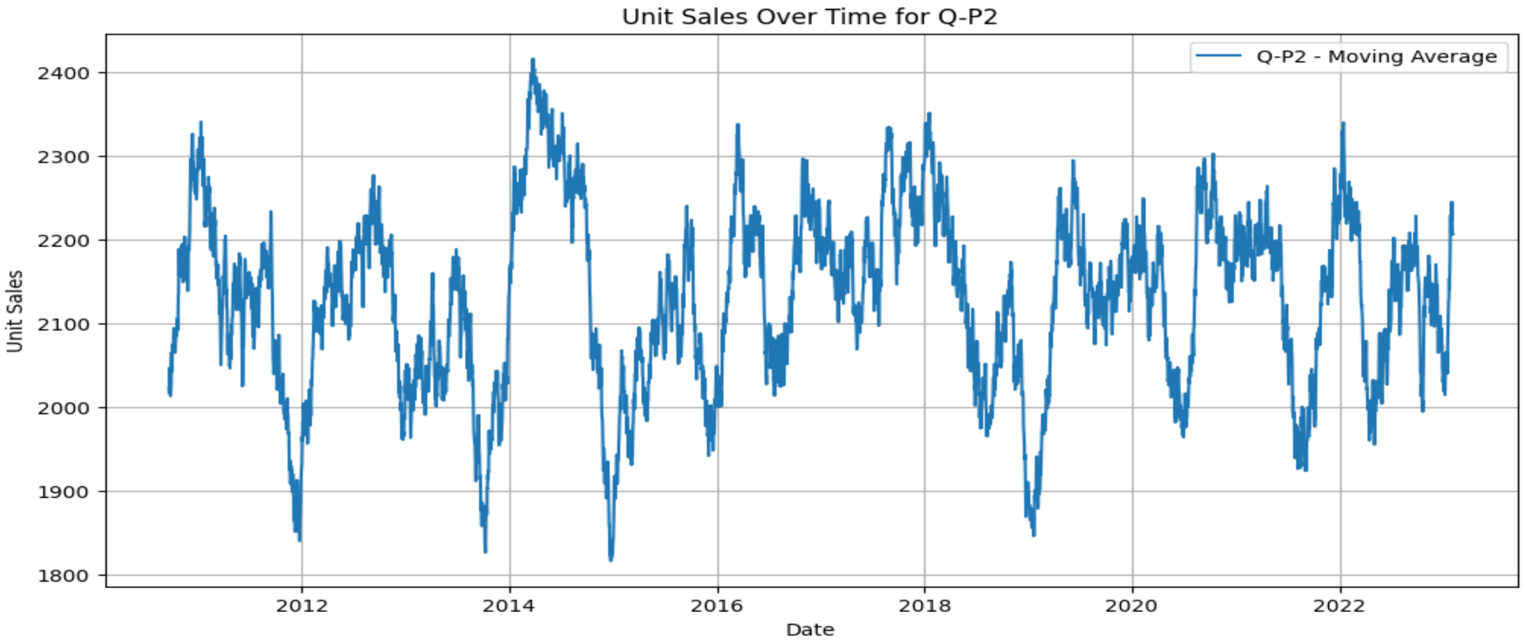
DATA VISUALIZATION

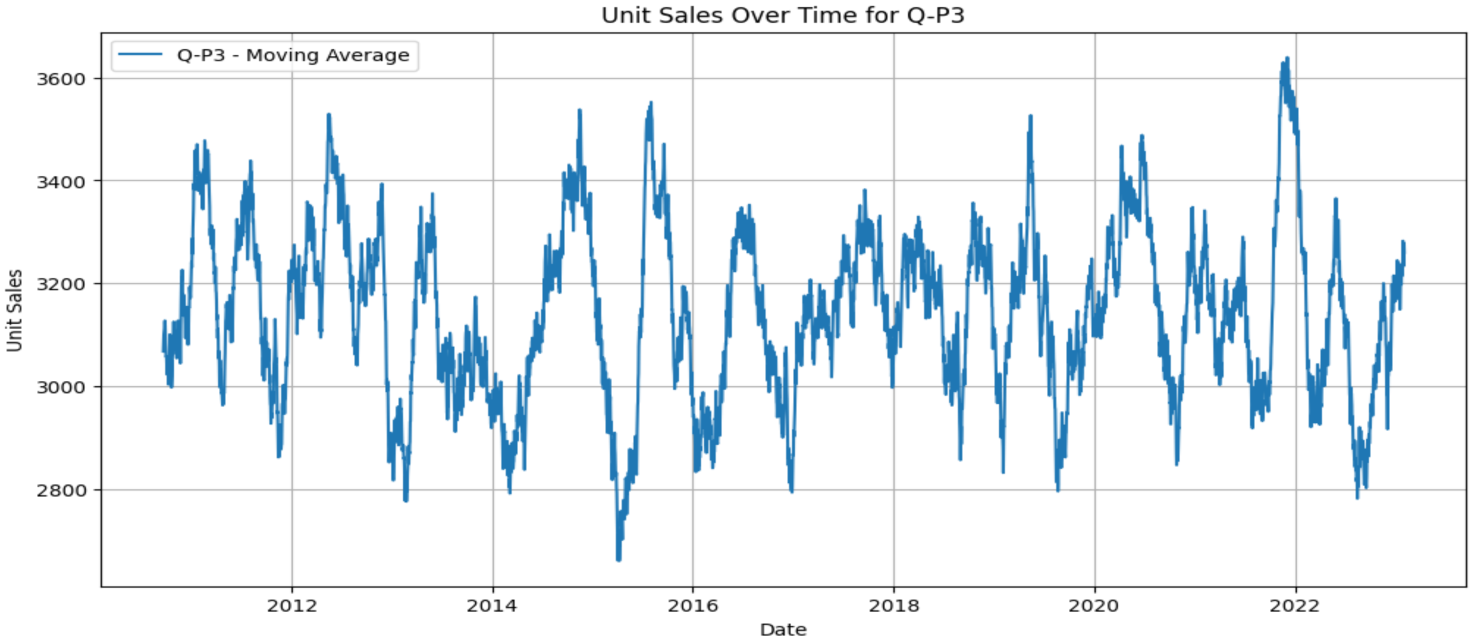
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
data = pd.read_csv('/kaggle/input/product-sales-data/statsfinal.csv‘
product_sales_df.shape
product_sales_df.columns
product_sales_df.head(10)
product_sales_df.dtypes
product_sales_df.isna().sum()
continuous_columns = ['S-P1', 'S-P2', 'S-P3', 'S-P4']
continuous_data_df = product_sales_df[continuous_columns]
continuous_data_df.head(5)
continuous_data_df.describe()
p1 = product_sales_df.drop(['Date','Unnamed: 0'], axis=1)
sns.pairplot(p1)
```

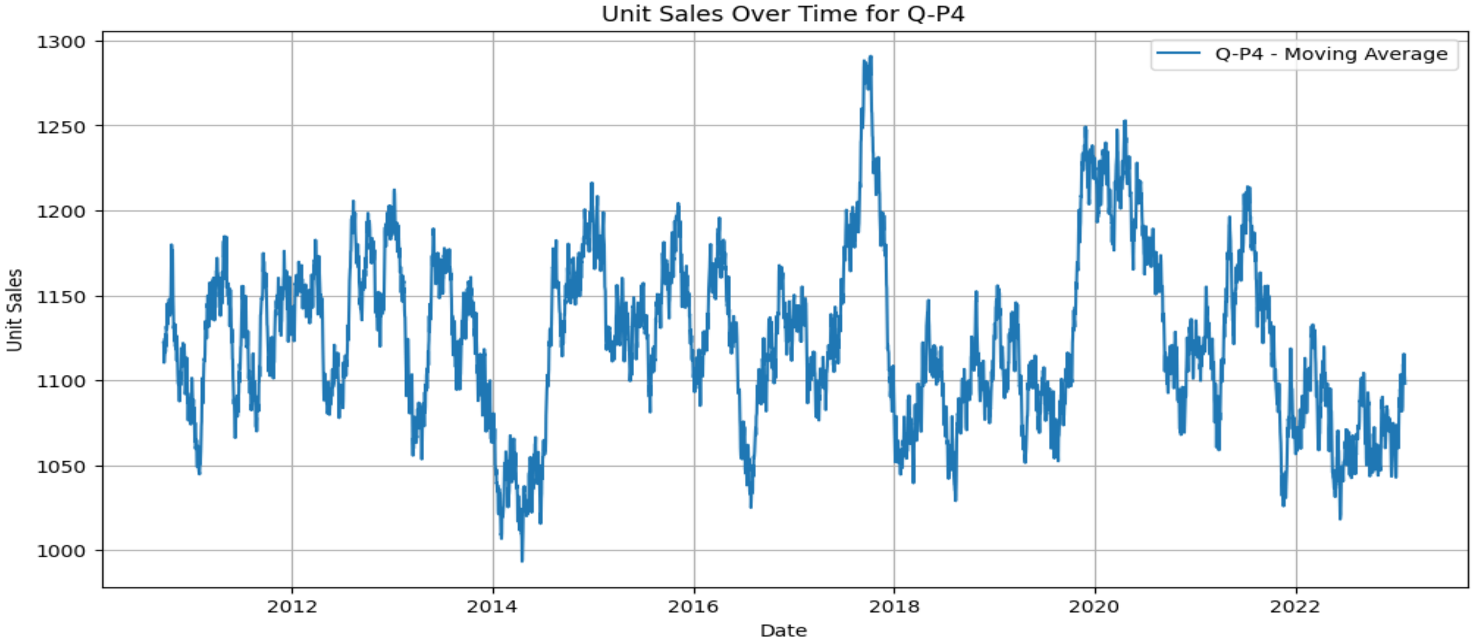

Q-P1
0
2000
4000
6000
8000
Q-P2
1000
2000
3000
4000
Q-P3
0
1000
2000
3000
4000
5000
6000
Q-P4
250
500
750
1000
1250
1500
1750
2000
S-P1
5000
10000
15000
20000
25000
S-P2
5000
10000
15000
20000
25000
S-P3
0
5000
10000
15000
20000
25000
30000
S-P4
2000
4000
6000
8000
10000
12000
14000
0 2000 4000 6000 8000
Q-P1
1000 2000 3000 4000
Q-P2
0 2000 4000 6000
Q-P3
500 1000 1500 2000
Q-P4
0 10000 20000
S-P1
10000 20000
S-P2
0 10000 20000 30000
S-P3
5000 10000
S-P4

```
df = product_sales_df
try:
    df['Date'] = pd.to_datetime(df['Date'], format='%d-%m-%Y', errors='coerce')
except pd.errors.ParserError:
    pass
df = df.dropna(subset=['Date'])
df['Date'] = pd.to_datetime(df['Date'], format='%d-%m-%Y')
df.set_index('Date', inplace=True)for product in products:
    product_data = df[product]
    moving_average = product_data.rolling(window=100).mean()
    plt.figure(figsize=(12, 6))
    plt.plot(df.index, moving_average, label=f'{product} - Moving Average')
    plt.xlabel('Date')
    plt.ylabel('Unit Sales')
    plt.title(f'Unit Sales Over Time for {product}')
    plt.grid(True)
    plt.legend()
    plt.show()
products = ['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']
```


Unit Sales Over Time for Q-P1
Q-P1 - Moving Average
Unit Sales
4600
4400
4200
4000
3800
3600
3400
2012
2014
2016
2018
2020
2022
Date


Unit Sales Over Time for Q-P2
Q-P2 - Moving Average
Unit Sales
2400
2300
2200
2100
2000
1900
1800
2012
2014
2016
2018
2020
2022
Date


Unit Sales Over Time for Q-P3
Q-P3 - Moving Average
Unit Sales
3600
3400
3200
3000
2800
2012
2014
2016
2018
2020
2022
Date


Unit Sales Over Time for Q-P4
Q-P4 - Moving Average
Unit Sales
1300
1250
1200
1150
1100
1050
1000
2012
2014
2016
2018
2020
2022
Date

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.
- Please note that this is a high-level overview, and the specific implementation details and choice of machine learning models may vary based on the characteristics of your dataset and the goals of your analysis.

THANK YOU