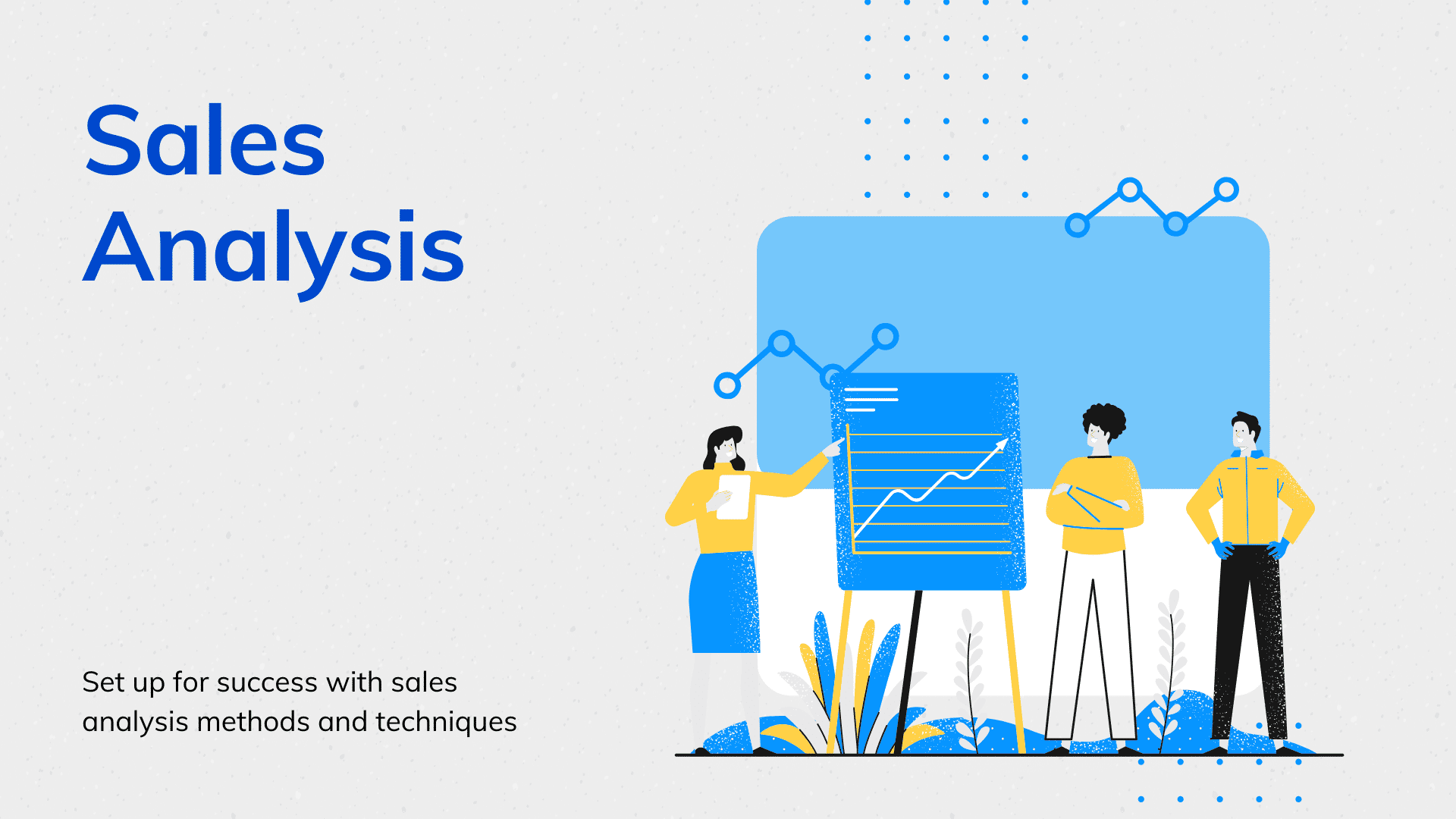
*PRODUCT SALES ANALYSIS*

Phase 5 submission document

**Project Title**

Product Sales Analysis



**Introduction**

In an increasingly competitive market, our organization recognizes the importance of analyzing project sales to enhance revenue, optimize resources, and improve customer satisfaction. This project aims to utilize the principles of Design Thinking to gain a deeper understanding of sales processes and identify innovative solutions for improving sales effectiveness and efficiency.

**Project Overview**

The project is about analyzing sales data. The dataset provided is a small-scaled business venture REC Corp LTD established in India. They have been selling Four products for over ten years. They have collected the information and organized it into a CSV file. The file includes information about the date of purchase of the products, the number of people who purchased each product and the revenue of each product.

**Data Source:**

A good data source should be accurate,reliable and consistent.

Dataset Link:( [Product Sales Data (kaggle.com)](https://www.kaggle.com/datasets/ksabishek/product-sales-data))

**Project Objective**

The objective involves extracting insights about top-selling products, peak sales periods and customer preferences. By understanding the sales trend and customer behavior we have to help businesses improve inventory and marketing strategies.

**Here's a list of tools and software commonly used in the process:**

**Programming Language:**

Python is the most popular language for machine learning due to its extensive libraries and frameworks. You can use libraries like NumPy, pandas, scikit-learn, and more.

**Integrated Development Environment (IDE)**

Choose an IDE for coding and running machine learning experiments. Some popular options include Jupyter Notebook, Google Colab, or traditional IDEs like PyCharm.

**Machine Learning Libraries:**

You'll need various machine learning libraries, including scikit-learn for building and evaluating machine learning models.- TensorFlow or PyTorch for deep learning, if needed.- XGBoost, LightGBM, or CatBoost for gradient boosting models.

**Data Visualization Tools**

Tools like Matplotlib, Seaborn, or Plotly are essential for data exploration and visualization.

**Data Preprocessing Tools**

Libraries like pandas help with data cleaning, manipulation, and preprocessing.

**Version Control**

Version control systems like Git are valuable for tracking changes in your code and collaborating with others.

**Notebooks and Documentation**

Tools for documenting your work, such as Jupyter Notebooks or Markdown for creating README files and documentation.

**Cloud-Based Deployment**

IBM Cognos Analytics Cloud is hosted in the cloud, which means that users can access it over the internet, eliminating the need for on-premises infrastructure. This cloud-based deployment allows for greater flexibility and scalability.

**Project Definition and Design Thinking**

**Empathize**

This involves understanding the data sources. In this project it is sales data and it contains 8 numerical parameters. They are as follows:

* Q1- Total unit sales of product 1
* Q2- Total unit sales of product 2
* Q3- Total unit sales of product 3
* Q4- Total unit sales of product 4
* S1- Total revenue from product 1
* S2- Total revenue from product 2
* S3- Total revenue from product 3
* S4- Total revenue from product 4

By empathizing with data in a product sales analysis, we can create a more holistic understanding of the data’s role in decision-making and problem-solving. This in turn can lead to more effective strategies and actions based on the insights derived from the data.

**Define: Problem Statement**

A problem statement is a concise and clear description of an issue or challenge that needs to be addressed or solved. In this project, the problem statement is analyzing the sales data and visualizing it.

**Ideate: Generating Creative Solutions**

Ideate is a crucial face that involves generating creative and innovative ideas to solve a problem or address a challenge. In the context of product sales analysis, the ideation phase aims to come up with fresh insights, strategies, or approaches to improve the analysis of product sales data and drive better business outcomes.

Some ideation techniques that can be used are

* DataVisualization

Explore innovative ways to visualize sales data. Consider interactive dashboards, heatmaps, 3D charts, or augmented reality interfaces to provide a more engaging and insightful view of sales performance.

* Predictive Analysis

Investigate the use of predictive modeling and machine learning to forecast future sales trends. How can advanced analytics techniques help in identifying patterns and making more accurate sales predictions?

**Prototype: Bringing Ideas to Life**

Creating prototypes is a crucial step in the design thinking process, as it allows you to bring your ideas to life, test them, and gather feedback before committing to full-scale implementation. In the context of product sales analysis, prototyping involves developing tangible representations of your proposed solutions or improvements to the sales analysis process. We have to select appropriate tools and software for creating the prototypes. Depending on the objectives, we might use spreadsheet software, data visualization tools, wireframing or mock-up tools, or even simple paper sketches. If the prototype involves visualizing sales data, create mock-ups or sample dashboards that illustrate how the data will be presented.

**Test**

Take your prototypes and put them in front of actual users. Collect feedback and iterate on your solutions. Testing allows you to refine your ideas and make necessary adjustments based on real user experiences.

**Implement**

Once you've tested and refined your ideas, it's time to move forward with implementing the final solution. This can involve developing a product, service, or process, and launching it into the real world.

**Evaluate**

Continuously monitor the performance of the machine learning model after implementation to ensure it remains accurate and relevant in a changing real estate market.

**Iterative Testing and Feedback**

Iterative testing and feedback are essential components of the design thinking process, especially when applied to product sales analysis. The iterative approach allows you to continually refine and improve your solutions by incorporating input from stakeholders, users, and data.

Iterative testing and feedback are not a one-time activity but an ongoing cycle of improvement. By continually refining the product sales analysis solutions based on user input and data-driven insights, we can ensure that the approach remains relevant and effective in a dynamic business environment.

**Scale and deploy**

Scalability Analysis: Assess the feasibility and scalability of the solution.

Resource Planning: Plan the resources, infrastructure, and processes needed for scaling.

Pilot Implementation: Implement the solution on a smaller scale to test its scalability and effectiveness.

Full**-**ScaleImplementation: Roll out the solution on a larger scale.

Monitoring and Evaluation: Continuously monitor the solution's performance and gather feedback.

Feedback Integration: Use ongoing feedback to make improvements and adjustments as needed.

**Educate and Train:**

Awareness Building: Start by creating awareness about design thinking within your organization. Explain the principles and benefits of the approach.

Training Workshops: Provide training sessions and workshops for employees to teach them the fundamentals of design thinking, including empathy, problem framing, ideation, and prototyping.

**Design into Innovation**

Consider incorporating machine learning algorithms to predict future sales trends or customer behaviors.



**Data Collection And Preprocessing:**

* Importing the dataset: Obtain a comprehensive dataset containing relevant features.
* Data collection: It is the process of gathering data for use in business decision making, strategic planning, research and other purposes. It helps improve services, understand consumer needs, refine business strategies, grow and retain customers, and even sell the data as second party data to other businesses at a profit.
* Data pre-processing: It is the concept of changing raw data into a clean dataset. The dataset is preprocessed in order to check missing values, noisy data, and other inconsistencies before executing it to the algorithm.

**Exploratory Data Analysis:**

* Visualize and analyse the dataset to gain insights into the relationships between variables.
* Identify correlations and patterns that can inform feature selection and engineering.
* Present various data visualizations to gain insights into the dataset.
* Explore correlations between features and the target variable
* Discuss any significant findings from the EDA phase that inform feature selection.

**Feature Engineering:**

* Create new features or transform existing ones to capture valuable information.
* Utilize domain knowledge to engineer features that may cause impact.
* Explain the process of creating new features or transforming existing ones.
* Showcase domain specific feature engineering such as proximity scores or composite indicators.
* Emphasize the impact of engineered features on model performance.

**Regression Techniques:**

* Linear Regression: **Linear regression is a powerful tool for understanding and predicting the behavior of a variable, however, it needs to meet a few conditions in order to be accurate and dependable solutions**
* Lasso Regression: Employ L1 regularization to perform feature selection and simplify the model.
* A Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. SVMs are particularly effective for solving binary classification problems, but they can also be adapted for multi-class classification and regression tasks
* Random Forest Regression: Implement an ensemble technique to handle nonlinearity and capture complex relationships in the data.
* Decision Tree:A decision tree is a supervised machine learning algorithm that is used for both classification and regression tasks.

**Forecasting:**

* Estimate the future revenue by predicting how much of a product or service will sell in the next week, month, quarter or year.
* Detail analysis of past and present trends or events to predict future events.
* Project the measure of how a market will respond to a company’s go to market efforts.
* A sales forecast must integrate a lot of data because the more important and qualified the data is the more accurate it will be.

**Customer Segmentation:**

* Create more specific sales and marketing strategies for customer groups.
* Discover insights that define specific segments of customers.
* Marketers and brands leverage this process to determine what campaigns, offers or products to leverage when communicating with specific segments.
* Divide a company’s customers into groups based on common characteristics so company’s can market to each other effectively and appropriately.
* Apply clustering techniques to segment customers based on their purchasing behaviour and tailor marketing techniques accordingly.

**Anomaly Detection:**

* Implement anomaly detection models to identify unusual sales patterns or fraudulent activities.
* Detect points on a given input time series where the behaviour isn’t what was expected or weird.
* Identify rare events, items, observations which are suspicious because they differ significantly from standard behaviours or patterns.

**Model Evaluation and Selection:**

* Split the dataset into training and testing sets.
* Evaluate models using appropriate metrics (e.g., Mean Absolute Error, Mean Squared Error, R-squared) to assess their performance.
* Use cross-validation techniques to tune hyper parameters and ensure model stability.
* Look at more insightful statistics of its performance.
* Decide what actions to take in order to improve this model.
* Compare the results with traditional linear regression models to highlight improvements.
* Select the best-performing model for further analysis.

**Model Interpretability:**

* Explain how to interpret feature importance from Gradient Boosting and XG Boost models.
* Discuss the insights gained from feature importance analysis and their relevance to product sales analysis.
* Interpret feature importance from ensemble models like Random Forest and Gradient Boosting to understand the factors influencing product sales.

**Business Insights:**

* Translate the machine learning results into actionable insights for the business which can inform price, inventory management and marketing strategies.

**Deployment Prediction:**

* Deploy the chosen regression model to predict product sales analysis.
* Develop a user-friendly interface for users to input property features and receive price predictions.

**Model 1- Linear Regression**

Import the necessary libraries:

import numpy as np

import pandas as pd

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

import matplotlib.pyplot as plt

Load your dataset into a Pandas DataFrame:

data=pd.read\_csv("C:\\Users\\jacin\\Downloads\\products\\statsfinal.csv")

Split the data into features (X) and the target variable (y):

X = data[['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']]

y = data['S-P1']

Split the data into training and testing sets:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Create a Linear Regression model and fit it to the training data

model = LinearRegression()

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

**Out: LinearRegression()**

Make predictions on the test set:

y\_pred = model.predict(X\_test)

Evaluate the model's performance:

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print(f'Mean Squared Error (MSE): {mse}')

print(f'R-squared (R2): {r2}')

**Out: Mean Squared Error (MSE): 1.139556492178734e-23**

**R-squared (R2): 1.0**

Create a scatter plot to visualize the actual vs. predicted values:

plt.scatter(y\_test, y\_pred)

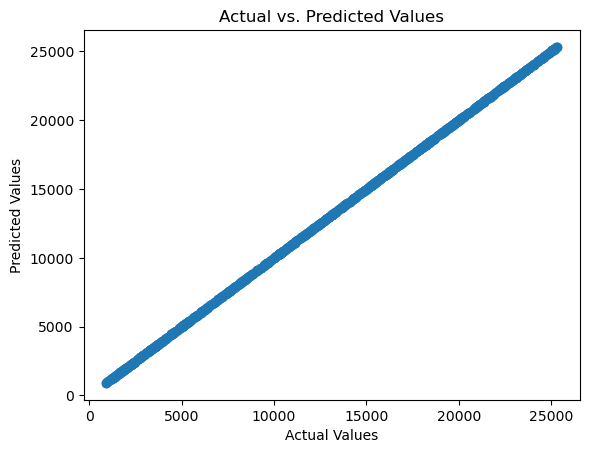
plt.xlabel('Actual Values')

plt.ylabel('Predicted Values')

plt.title('Actual vs. Predicted Values')

plt.show()

**Out**:



You can also visualize the coefficients of the model to understand

the feature importance:

coefficients = model.coef\_

feature\_names = X.columns

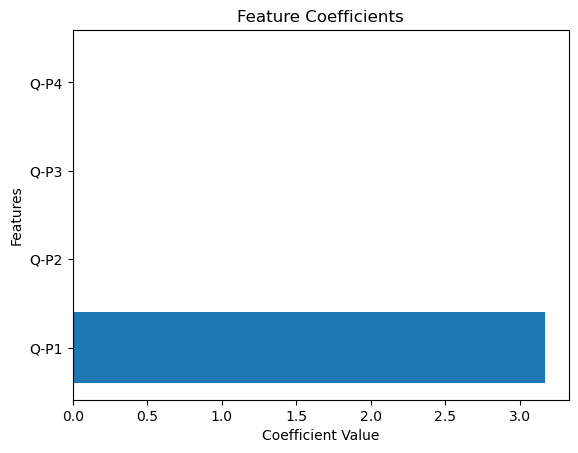
plt.barh(feature\_names, coefficients)

plt.xlabel('Coefficient Value')

plt.ylabel('Features')

plt.title('Feature Coefficients')

plt.show()

**Out:**

**Model 2: Lasso Regression**

Import the necessary libraries:

import numpy as np

import pandas as pd

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

from sklearn.linear\_model import Lasso

from sklearn.metrics import mean\_squared\_error, r2\_score

import matplotlib.pyplot as plt

Load your dataset into a Pandas DataFrame:

data=pd.read\_csv("C:\\Users\\jacin\\Downloads\\products\\statsfinal.csv")

Split the data into features (X) and the target variable (y). In your case, you want to predict 'S-P1' (Total revenue from product 1) based on the other columns (Q-P1, Q-P2, Q-P3, Q-P4, S-P2, S-P3, S-P4).

X = data[['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']]

y = data['S-P1']

Split the data into training and testing sets:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Create a Lasso Regression model and fit it to the training data:

alpha = 1.0

lasso\_model = Lasso(alpha=alpha)

lasso\_model.fit(X\_train, y\_train)

**Out: Lasso()**

Make predictions on the test set:

y\_pred = lasso\_model.predict(X\_test)

Evaluate the model's performance:

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print(f'Mean Squared Error (MSE): {mse}')

print(f'R-squared (R2): {r2}')

**Out: Mean Squared Error (MSE): 2.016327108852503e-07**

**R-squared (R2): 0.999999999999996**

Create a scatter plot to visualize the actual vs. predicted values:

plt.scatter(y\_test, y\_pred)

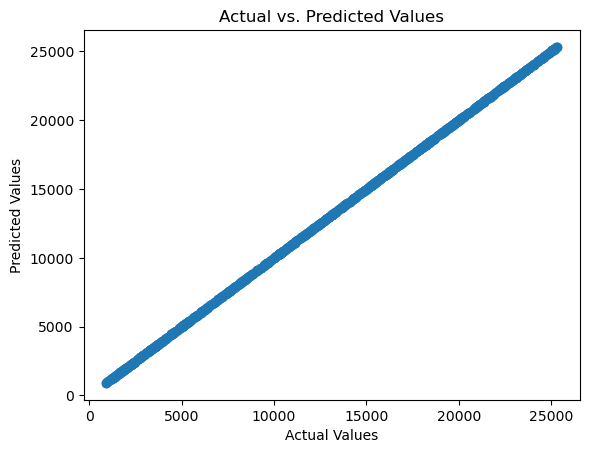
plt.xlabel('Actual Values')

plt.ylabel('Predicted Values')

plt.title('Actual vs. Predicted Values')

plt.show()

**Out:**



You can also visualize the coefficients of the model to understand the feature importance:

coefficients = lasso\_model.coef\_

feature\_names = X.columns

plt.barh(feature\_names, coefficients)

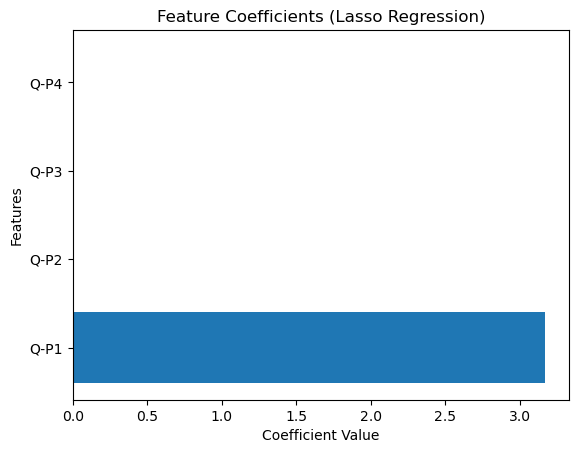
plt.xlabel('Coefficient Value')

plt.ylabel('Features')

plt.title('Feature Coefficients (Lasso Regression)')

plt.show()

**Out:**



**Model 3: Support Vector machine**

Import the necessary libraries:

import numpy as np

import pandas as pd

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

from sklearn.svm import SVR

from sklearn.metrics import mean\_squared\_error, r2\_score

import matplotlib.pyplot as plt

Load your dataset into a Pandas DataFrame:

data=pd.read\_csv("C:\\Users\\jacin\\Downloads\\products\\statsfinal.csv")

Split the data into features (X) and the target variable (y). In your case, you want to predict 'S-P1' (Total revenue from product 1) based on the other columns (Q-P1, Q-P2, Q-P3, Q-P4, S-P2, S-P3, S-P4

X = data[['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4', 'S-P2', 'S-P3', 'S-P4']]

y = data['S-P1']

Split the data into training and testing sets:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Create a Support Vector Machine Regression model and fit it to the training data:

svm\_model = SVR(kernel='linear')

svm\_model.fit(X\_train, y\_train)

**Out: SVR(kernel='linear')**

Make predictions on the test set:

y\_pred = svm\_model.predict(X\_test)

Evaluate the model's performance:

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print(f'Mean Squared Error (MSE): {mse}')

print(f'R-squared (R2): {r2}')

**Out: Mean Squared Error (MSE): 0.0015957374427032214**

**R-squared (R2): 0.9999999999687661**

Create a scatter plot to visualize the actual vs. predicted values:

plt.scatter(y\_test, y\_pred)

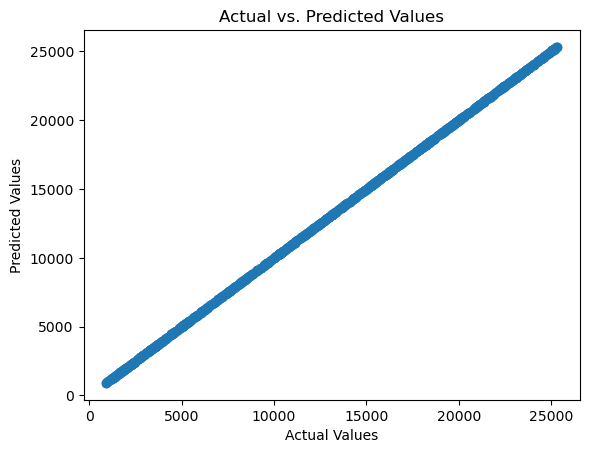
plt.xlabel('Actual Values')

plt.ylabel('Predicted Values')

plt.title('Actual vs. Predicted Values')

plt.show()

**Out:**



**Model 4: Decision tree**

Import the necessary libraries:

import numpy as np

import pandas as pd

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

from sklearn.tree import DecisionTreeRegressor

from sklearn.metrics import mean\_squared\_error, r2\_score

import matplotlib.pyplot as plt

Load your dataset into a Pandas DataFrame:

data=pd.read\_csv("C:\\Users\\jacin\\Downloads\\products\\statsfinal.csv")

Split the data into features (X) and the target variable (y):

X = data[['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']]

y = data['S-P1']

Split the data into training and testing sets:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Create a Decision Tree Regressor model and fit it to the training data:

tree\_model = DecisionTreeRegressor(random\_state=42)

tree\_model.fit(X\_train, y\_train)

**Out:DecisionTreeRegressor(random\_state=42)**

Make predictions on the test set:

y\_pred = tree\_model.predict(X\_test)

Evaluate the model's performance:

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print(f'Mean Squared Error (MSE): {mse}')

print(f'R-squared (R2): {r2}')

**Out: Mean Squared Error (MSE): 107.02078499999966**

**R-squared (R2): 0.9999979052497234**

Create a scatter plot to visualize the actual vs. predicted values:

plt.scatter(y\_test, y\_pred)

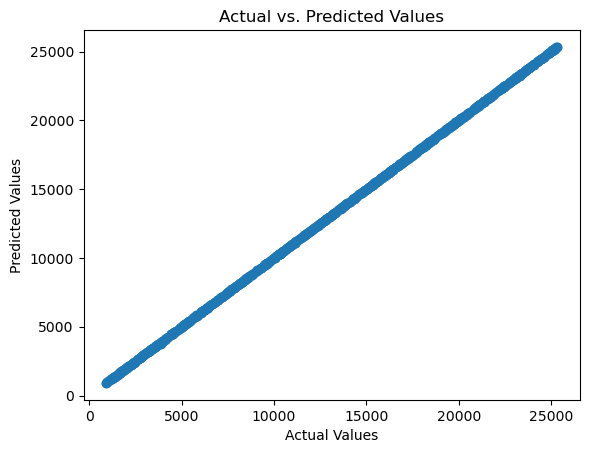
plt.xlabel('Actual Values')

plt.ylabel('Predicted Values')

plt.title('Actual vs. Predicted Values')

plt.show()

**Out:**



**Model 5: Random Forest**

Import the necessary libraries:

import numpy as np

import pandas as pd

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

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_squared\_error, r2\_score

import matplotlib.pyplot as plt

Load your dataset into a Pandas DataFrame:

data=pd.read\_csv("C:\\Users\\jacin\\Downloads\\products\\statsfinal.csv")

Split the data into features (X) and the target variable (y):

X = data[['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']]

y = data['S-P1']

Split the data into training and testing sets:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Create a Random Forest Regressor model and fit it to the training data:

rf\_model = RandomForestRegressor(n\_estimators=100, random\_state=42) # You can adjust the number of trees (n\_estimators)

rf\_model.fit(X\_train, y\_train)

**Out: RandomForestRegressor(random\_state=42)**

Make predictions on the test set:

y\_pred = rf\_model.predict(X\_test)

Evaluate the model's performance:

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print(f'Mean Squared Error (MSE): {mse}')

print(f'R-squared (R2): {r2}')

**Out: Mean Squared Error (MSE): 29.94948159932641**

**R-squared (R2): 0.9999994137897151**

Create a scatter plot to visualize the actual vs. predicted values:

plt.scatter(y\_test, y\_pred)

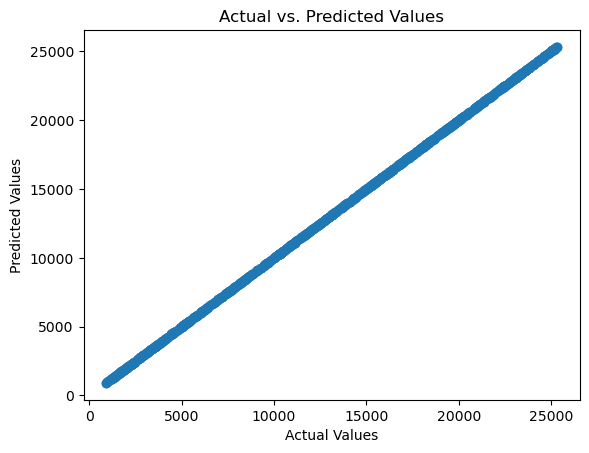
plt.xlabel('Actual Values')

plt.ylabel('Predicted Values')

plt.title('Actual vs. Predicted Values')

plt.show()

**Out:**



You can also investigate feature importance using the feature\_importances\_ attribute of the Random Forest model

feature\_importance = rf\_model.feature\_importances\_

feature\_names = ['Q1', 'Q2', 'Q3', 'Q4']

plt.barh(feature\_names, feature\_importance)

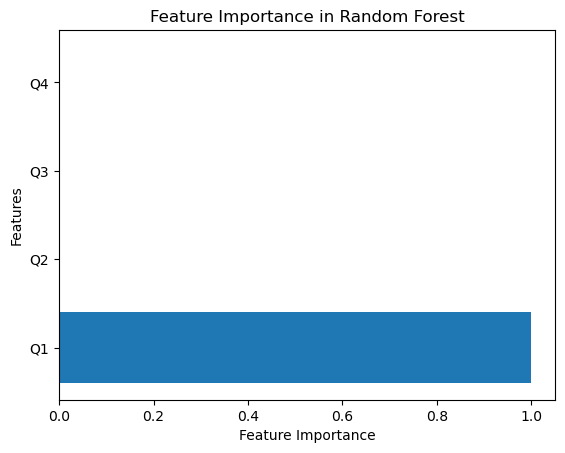
plt.xlabel('Feature Importance')

plt.ylabel('Features')

plt.title('Feature Importance in Random Forest')

plt.show()

**Out:**



**Build Loading and Preprocessing the Dataset**

**Development Part 1**

**Processing and cleaning the data**

Importing the library and loading the dataset

import pandas as pd

import matplotlib.pyplot as plt

data=pd.read\_csv("C:\\Users\\jacin\\Downloads\\products\\statsfinal.csv")

print(data.to\_string())

**Out:**

Unnamed: 0 Date Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 S-P3 S-P4

0 0 13-06-2010 5422 3725 576 907 17187.74 23616.50 3121.92 6466.91

1 1 14-06-2010 7047 779 3578 1574 22338.99 4938.86 19392.76 11222.62

2 2 15-06-2010 1572 2082 595 1145 4983.24 13199.88 3224.90 8163.85

3 3 16-06-2010 5657 2399 3140 1672 17932.69 15209.66 17018.80 11921.36

4 4 17-06-2010 3668 3207 2184 708 11627.56 20332.38 11837.28 5048.04

5 5 18-06-2010 2898 2539 311 1513 9186.66 16097.26 1685.62 10787.69

6 6 19-06-2010 6912 1470 1576 1608 21911.04 9319.80 8541.92 11465.04

7 7 20-06-2010 5209 2550 3415 842 16512.53 16167.00 18509.30 6003.46

8 8 21-06-2010 6322 852 3646 1377 20040.74 5401.68 19761.32 9818.01

9 9 22-06-2010 6865 414 3902 562 21762.05 2624.76 21148.84 4007.06

10 10 23-06-2010 1287 3955 2710 1804 4079.79 25074.70 14688.20 12862.52

11 11 24-06-2010 2197 1429 2754 1299 6964.49 9059.86 14926.68 9261.87

12 12 25-06-2010 7910 1622 5574 306 25074.70 10283.48 30211.08 2181.78

13 13 26-06-2010 3855 1015 1746 608 12220.35 6435.10 9463.32 4335.04

14 14 27-06-2010 5988 3288 916 1530 18981.96 20845.92 4964.72 10908.90

15 15 28-06-2010 2653 1544 3867 652 8410.01 9788.96 20959.14 4648.76

16 16 29-06-2010 3664 2294 3244 897 11614.88 14543.96 17582.48 6395.61

17 17 30-06-2010 7077 2297 5376 1130 22434.09 14562.98 29137.92 8056.90

18 18 01-07-2010 3509 700 1175 1205 11123.53 4438.00 6368.50 8591.65

19 19 02-07-2010 3716 3175 651 1263 11779.72 20129.50 3528.42 9005.19

20 20 03-07-2010 7746 2883 671 728 24554.82 18278.22 3636.82 5190.64

21 21 04-07-2010 7006 2833 758 1005 22209.02 17961.22 4108.36 7165.65

print(data.describe())

**Out:**

Unnamed: 0 Q-P1 Q-P2 Q-P3 Q-P4 \

count 4600.000000 4600.000000 4600.000000 4600.000000 4600.000000

mean 2299.500000 4121.849130 2130.281522 3145.740000 1123.500000

std 1328.049949 2244.271323 1089.783705 1671.832231 497.385676

min 0.000000 254.000000 251.000000 250.000000 250.000000

25% 1149.750000 2150.500000 1167.750000 1695.750000 696.000000

50% 2299.500000 4137.000000 2134.000000 3202.500000 1136.500000

75% 3449.250000 6072.000000 3070.250000 4569.000000 1544.000000

max 4599.000000 7998.000000 3998.000000 6000.000000 2000.000000

S-P1 S-P2 S-P3 S-P4

count 4600.000000 4600.000000 4600.000000 4600.000000

mean 13066.261743 13505.984848 17049.910800 8010.555000

std 7114.340094 6909.228687 9061.330694 3546.359869

min 805.180000 1591.340000 1355.000000 1782.500000

25% 6817.085000 7403.535000 9190.965000 4962.480000

50% 13114.290000 13529.560000 17357.550000 8103.245000

75% 19248.240000 19465.385000 24763.980000 11008.720000

max 25353.660000 25347.320000 32520.000000 14260.000000

print(data.info())

**Out:**

<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

As we can see that there are 4600 entries and all entries are non null.

Accuracy of the data

# Step 1: Check for missing values

missing\_values = data.isnull().sum()

print("Missing Values:")

print(missing\_values)

# Step 2: Check for duplicates

duplicate\_rows = data[data.duplicated()]

print("Duplicate Rows:")

print(duplicate\_rows)

# Step 3: Verify data consistency and accuracy

# Calculate expected revenues based on unit sales and compare with actual revenues

for i in range(1, 5):

expected\_revenue = data[f'Q-P{i}'] \* data[f'S-P{i}']

actual\_revenue = data[f'S-P{i}']

discrepancy = expected\_revenue - actual\_revenue

discrepancy\_percentage = (discrepancy / expected\_revenue) \* 100

print(f"Discrepancy for Product {i}:")

print(discrepancy\_percentage)

# Step 4: Visualizations for further analysis

# For example, you can create scatter plots to visualize the relationship between unit sales and revenue for each product

for i in range(1, 5):

plt.scatter(data[f'Q-P{i}'], data[f'S-P{i}'], label=f'Product {i}')

plt.xlabel("Unit Sales")

plt.ylabel("Revenue")

plt.legend()

plt.show()

**Out:**

Missing Values:

Unnamed: 0 0

Date 0

Q-P1 0

Q-P2 0

Q-P3 0

Q-P4 0

S-P1 0

S-P2 0

S-P3 0

S-P4 0

dtype: int64

Duplicate Rows:

Empty DataFrame

Columns: [Unnamed: 0, Date, Q-P1, Q-P2, Q-P3, Q-P4, S-P1, S-P2, S-P3, S-P4]

Index: []

Discrepancy for Product 1:

0 99.981557

1 99.985810

2 99.936387

3 99.982323

4 99.972737

...

4595 99.959612

4596 99.986570

4597 99.984099

4598 99.967969

4599 99.918963

Length: 4600, dtype: float64

Discrepancy for Product 2:

0 99.973154

1 99.871630

2 99.951969

3 99.958316

4 99.968818

...

4595 99.970752

4596 99.881094

4597 99.968183

4598 99.915825

4599 99.974053

Length: 4600, dtype: float64

Discrepancy for Product 3:

0 99.826389

1 99.972051

2 99.831933

3 99.968153

4 99.954212

...

4595 99.809524

4596 99.979275

4597 99.972129

4598 99.983048

4599 99.956915

Length: 4600, dtype: float64

Discrepancy for Product 4:

0 99.889746

1 99.936468

2 99.912664

3 99.940191

4 99.858757

...

4595 99.926416

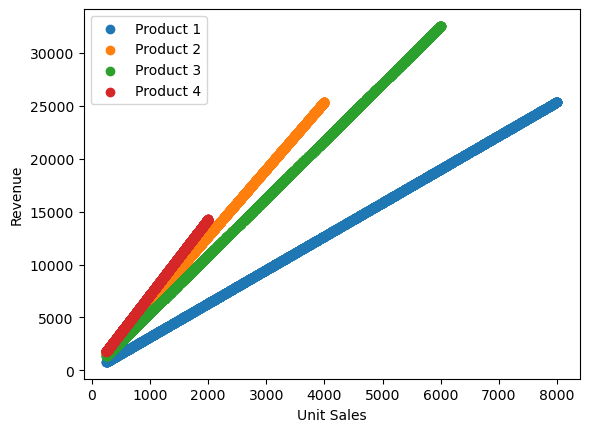
4596 99.923722

4597 99.789030

4598 99.806576

4599 99.753695

Length: 4600, dtype: float64



**Using IBM Cognos for visualization**

IBM Cognos provides a robust reporting and dashboarding system. Users can create various types of reports and dashboards to present data in a visual format. These reports and dashboards can include charts, graphs, tables, and other visual elements to convey information effectively. Cognos can connect to various data sources, including relational databases, spreadsheets, data warehouses, and cloud-based data stores. This flexibility enables users to access and analyze data from multiple sources in one central location.

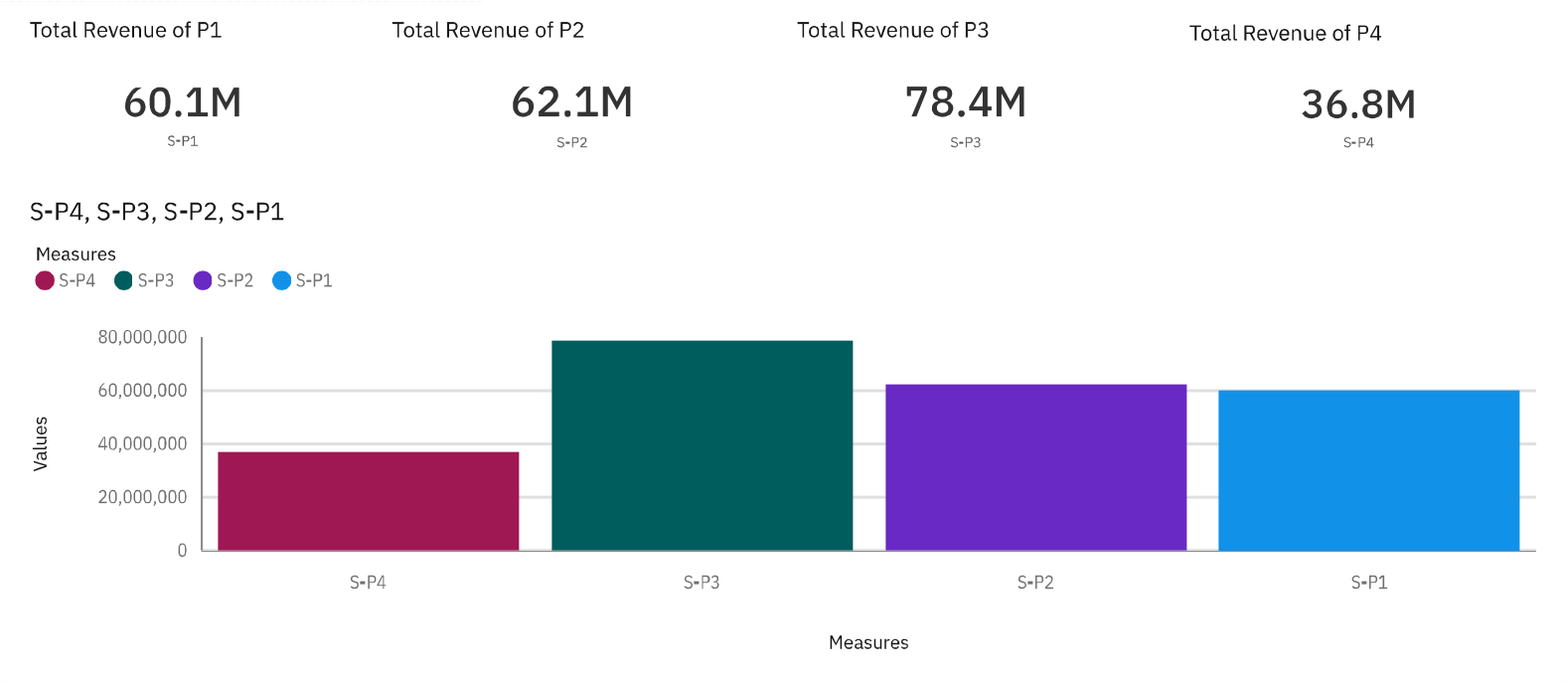
Cognos includes tools for data modeling, which helps users structure and organize data for reporting and analysis. Data modeling is essential for creating meaningful and insightful visualizations.

Users can interact with data and create ad-hoc reports and dashboards. They can drill down into data to discover more details and make data-driven decisions in real-time.

**Product with the most sales**

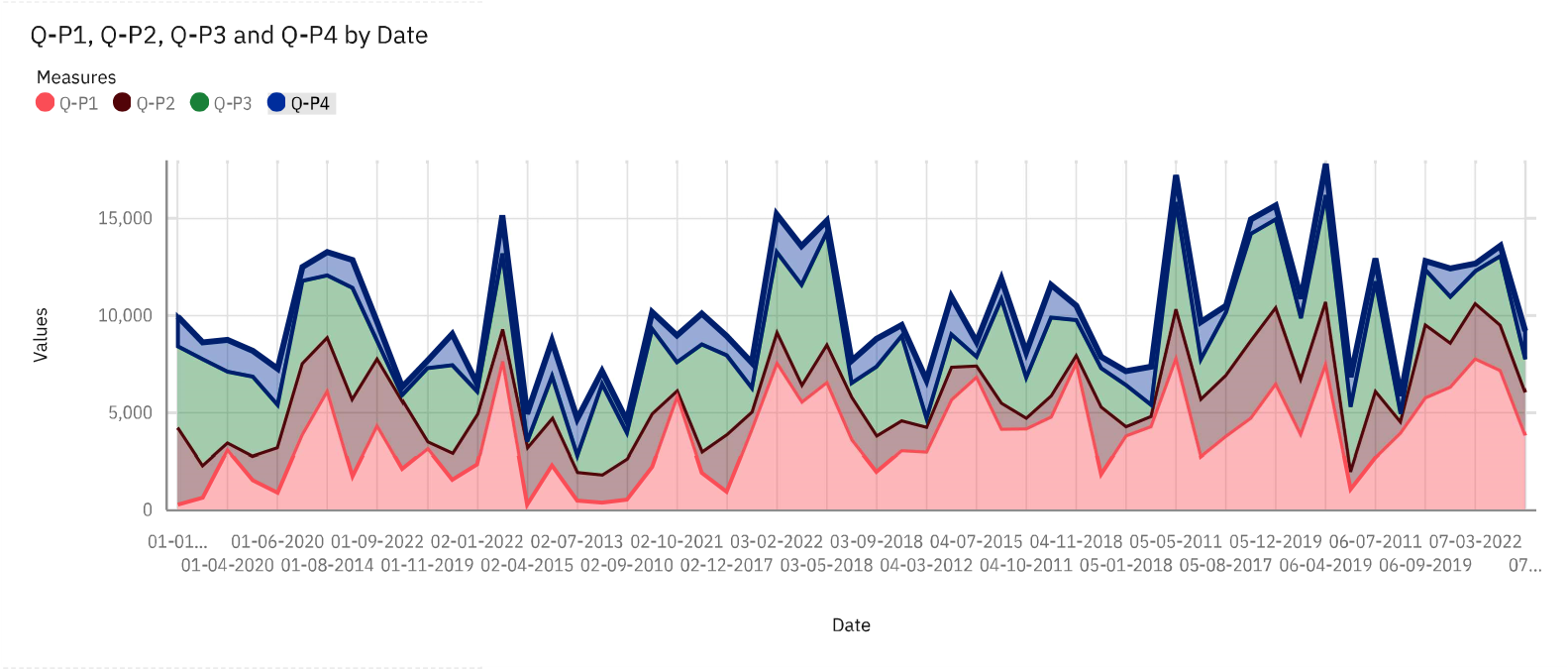
The sales S-P1,S-P2,S-P3 and S-P4 are given . From this we have to find the product with the most sales . S-P1 has a total revenue of 60.1M ,S-P2 has a total revenue of 62.1 M , S-P3 has a total revenue of 78.4M and S-P4 has a total revenue of 36.8 M . This is visualized in the form of a bar graph.

As we can see from the graph, product 3 has the most sales.



**Insights about the Purchase**

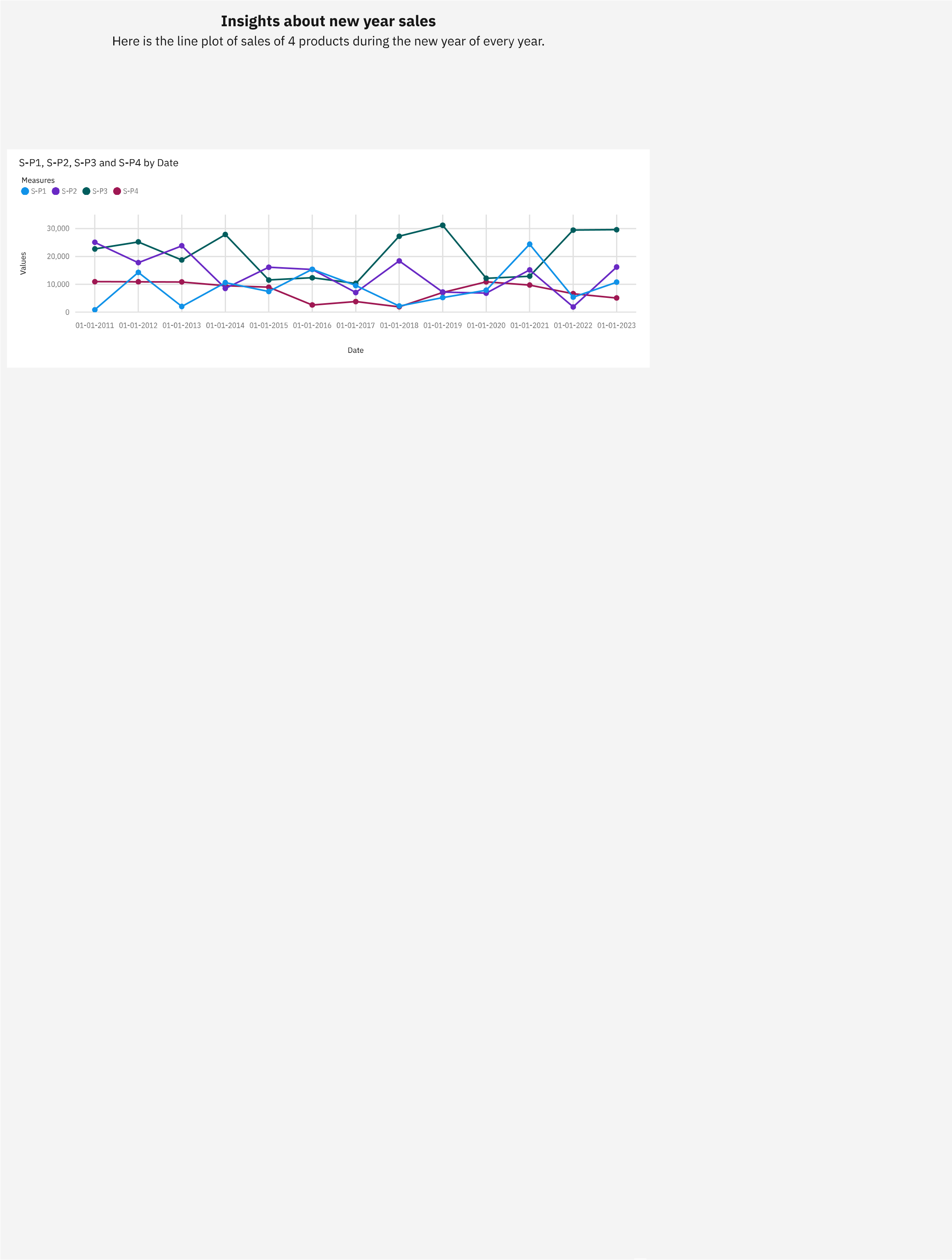
This is the area graph of the No of Customers of each product on randomly selected dates. Wherever the graph peaks it means that many customers purchased the product.



**Insights about new year sales**

Here is the lineplot of sales of 4 products during the new year of every year.

"New Year Sales" refers to promotional events and discounts offered by retailers and businesses at the beginning of a new year, typically in January. These sales are designed to attract customers and boost sales following the holiday season and to help clear out excess inventory.



**Insights about Product**

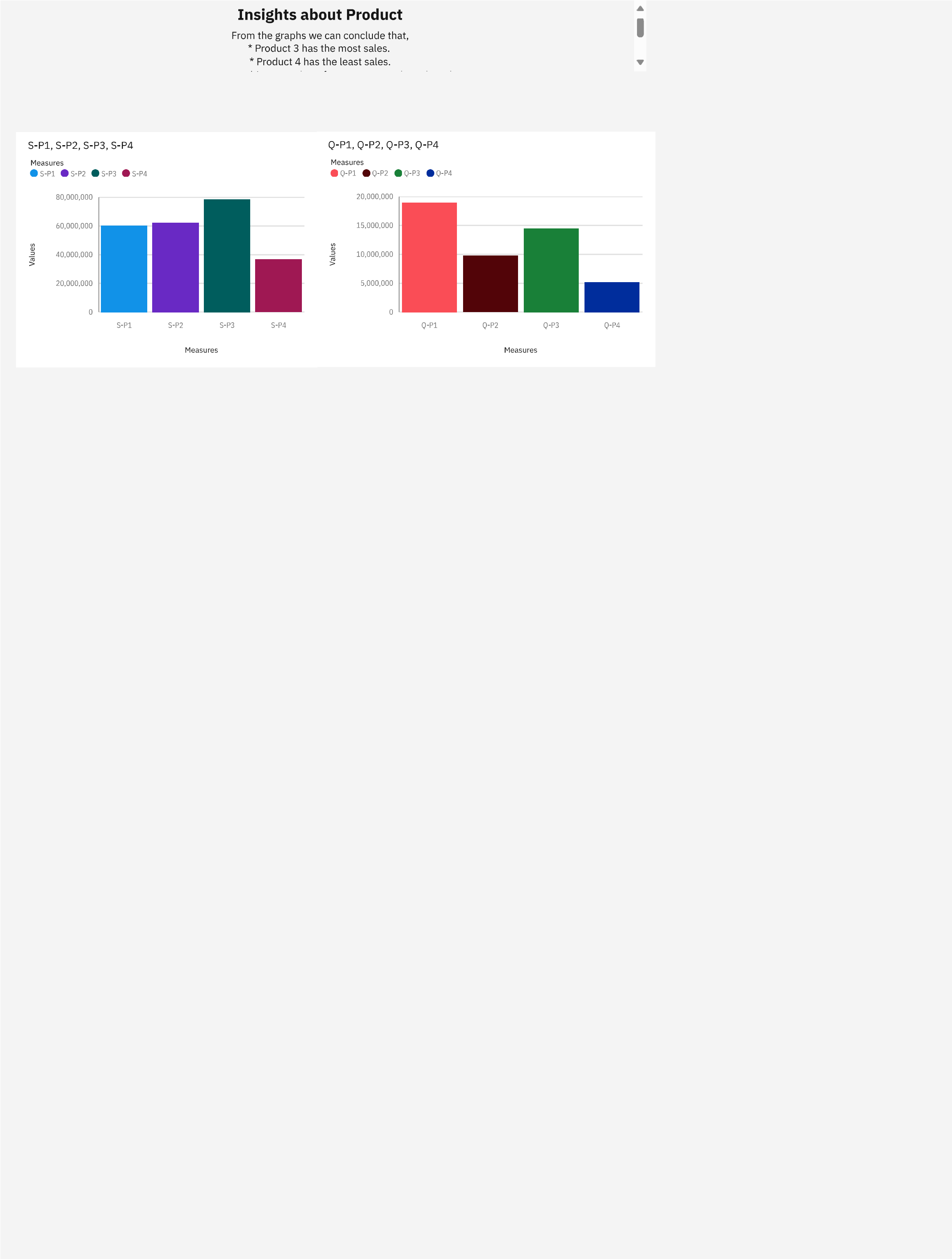
From the graphs we can conclude that,

\* ﻿Product 3 has the most sales.

\* Product 4 has the least sales.

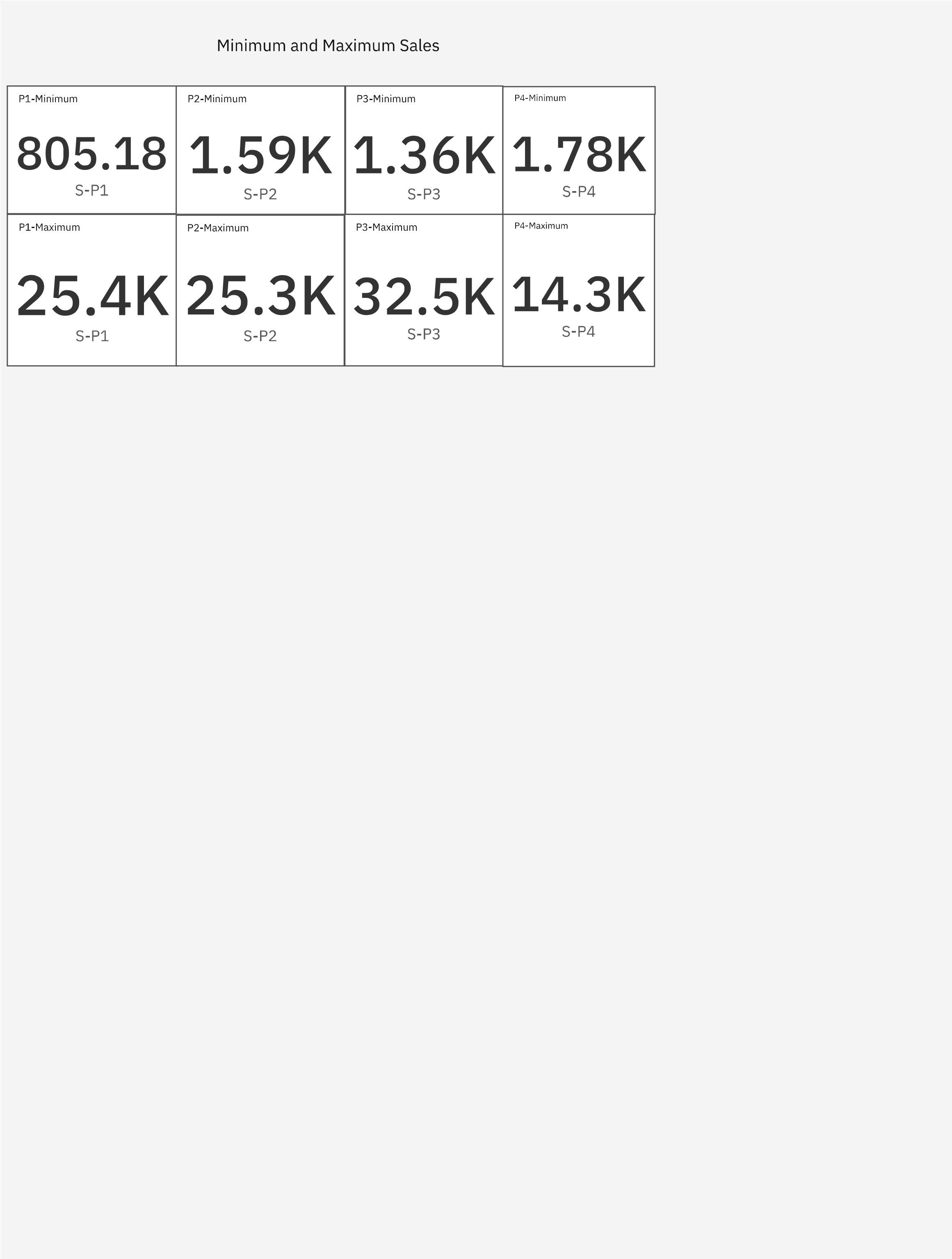
\* Less number of customers purchased Product 4.

So we can conclude that Product-4 was not purchased much and it has less sales.



**Minimum and Maximum Sales**

Minimum and maximum sales insights are essential aspects of data analysis, particularly in business and sales contexts. They provide valuable information about the range and extremes of sales data, helping organizations make informed decisions, set benchmarks, and identify outliers.

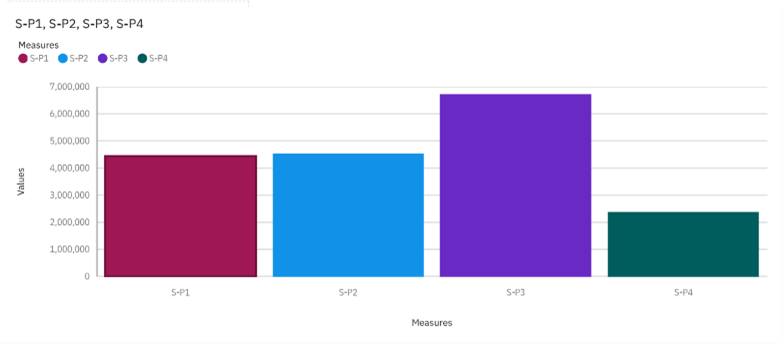


**Continue building your project using IBM Cognos**

**Development Part 2**

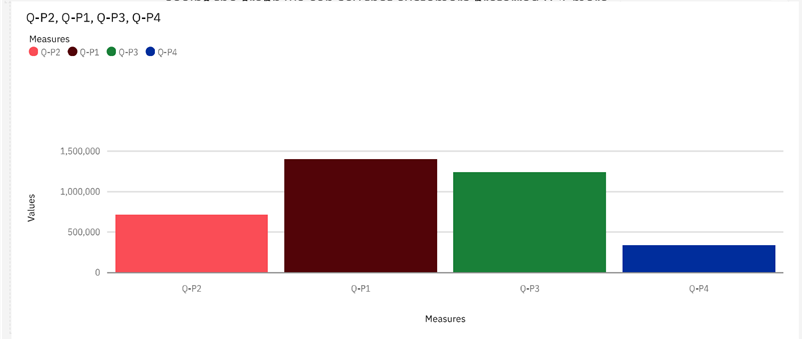
**Top Selling Product**

The top-selling product reflects strong market demand. It indicates that customers find this product valuable and are willing to purchase it, which can be due to factors like usefulness, quality, or brand reputation. By looking at the graph we can say that S-P3 is the most selling product.



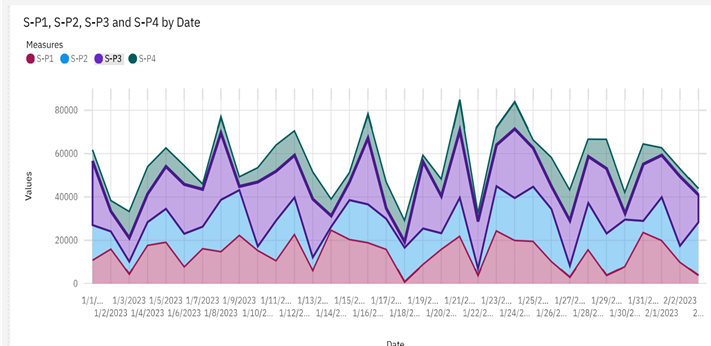
**Customer Preferences**

Customer preferences can vary widely depending on the industry, target audience, and cultural factors, but there are some general insights into customer preferences that can apply to a wide range of businesses. Byseeing the graph we can say that customers preferred P-1 more.



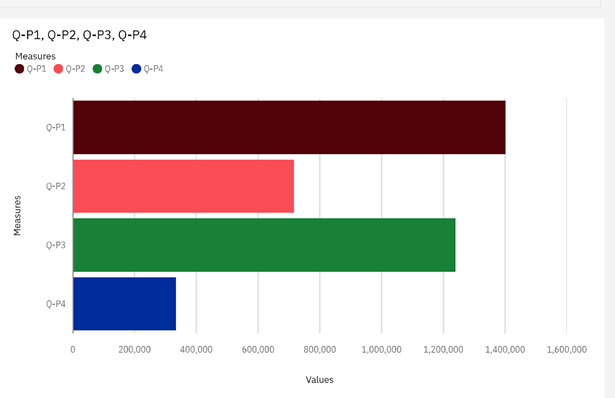
**December month sales for the year 2023**

This is the area graph for the last month of the year(2023).December is typically a significant month for sales across various industries, and it's especially noteworthy due to the holiday season and end-of-year shopping.



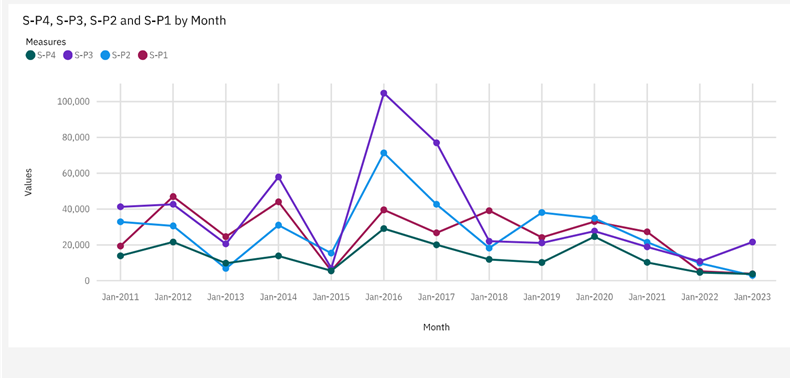
**Customer Preferences for specific products**

* Customers clearly favor product 1 over the other products.
* The majority of customers expressed higher satisfaction with product 1 compared to products 2,3 and 4.
* products 1 features are highly valued by the customers.
* Customer feedback indicates that product 2,3 and 4 lag behind product 1 in terms of price and performance.



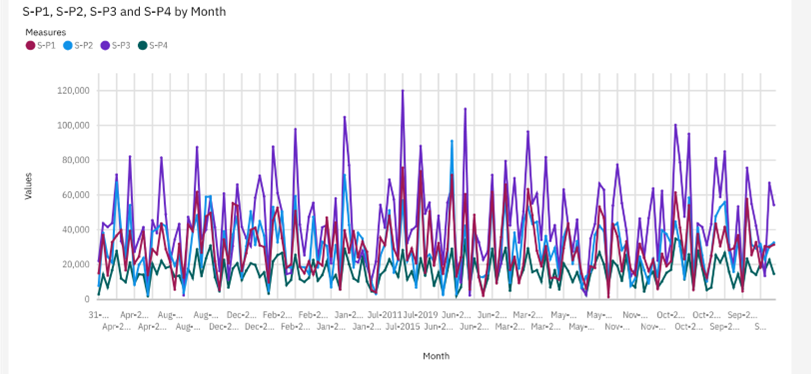
**January month sales**

﻿This is the January month sales for all the years given. It's a month that marks transitions and new beginnings, both on a personal and societal level. We can see that the sales were at peak during January 2016, especially for product 3.



**Sales Trends**

This is the lineplot for sales trends of S1,S2,S3,S4 by month.



**Advantages**

1. Performance Evaluation: Sales analysis provides a clear picture of how well each product is performing. It helps identify top-performing and underperforming products, enabling businesses to allocate resources more effectively.
2. Revenue Insights: By analyzing product sales, businesses can determine which products contribute the most to their revenue. This information can guide pricing strategies, marketing efforts, and inventory management.
3. Demand Forecasting: Sales analysis helps businesses anticipate product demand. This is crucial for inventory management, ensuring that there is enough stock to meet customer demand without overstocking and incurring storage costs.
4. Inventory Optimization: Analyzing sales data helps businesses maintain optimal inventory levels. This reduces carrying costs, minimizes the risk of overstocking or stockouts, and improves cash flow.
5. Pricing Strategy: By examining sales data, businesses can evaluate the effectiveness of their pricing strategies. They can adjust prices based on demand elasticity, competition, and customer preferences to maximize profitability.
6. Marketing Effectiveness: Businesses can measure the impact of marketing campaigns and promotional activities on product sales. This insight allows for the refinement of marketing efforts to achieve better ROI.
7. Product Development: Sales analysis can inform product development decisions. It helps identify customer preferences, allowing businesses to create new products or improve existing ones to better align with market demands.
8. Seasonal Trends: Understanding sales data over time reveals seasonal trends and patterns. This knowledge allows businesses to prepare for seasonal fluctuations and tailor marketing and inventory strategies accordingly.
9. Customer Segmentation: Sales data analysis can identify different customer segments based on their purchasing behavior. Businesses can then target these segments with tailored marketing and product offerings.
10. Market Expansion: By analyzing sales data, businesses can identify growth opportunities in new markets or demographics and adjust their sales and marketing strategies accordingly.

**Disadvantages**

While product sales analysis offers many advantages, it also comes with certain disadvantages and limitations that businesses should be aware of. Here are some of the potential disadvantages of product sales analysis:

~ Data Complexity: Analyzing sales data can be complex, particularly for businesses with a wide range of products and sales channels. Managing and processing large volumes of data can be resource-intensive and time-consuming.

~ Data Accuracy: The quality of sales data is crucial for meaningful analysis. Inaccurate or incomplete data can lead to incorrect conclusions and poor decision-making. Businesses need to invest in data quality assurance and validation.

~ Subjectivity: Interpreting sales data may involve some level of subjectivity. Different analysts may draw different conclusions from the same data, which can lead to disagreements and biases.

~ Historical Focus: Sales analysis typically looks at past data, which may not always predict future market trends accurately. Past performance doesn't always guarantee future success, especially in dynamic markets.

**Benefits**

Using cloud-based technology for product sales analysis offers several benefits to businesses, regardless of their size or industry. Here are some of the advantages of leveraging the cloud for sales analysis:

1. Scalability: Cloud-based solutions allow businesses to scale their sales analysis infrastructure easily. Whether your data volume increases or decreases, you can adapt your cloud resources accordingly, avoiding the need for large capital investments in on-premises hardware.
2. Cost-Efficiency: Cloud services operate on a pay-as-you-go model, which can be more cost-effective than purchasing and maintaining physical hardware and software licenses. This model allows businesses to manage their budgets more effectively.
3. Accessibility: Cloud-based sales analysis can be accessed from anywhere with an internet connection. This flexibility enables remote work and collaboration among team members in different locations.
4. Real-Time Data: Cloud platforms often offer real-time data processing and analysis capabilities. This means businesses can make more immediate and informed decisions based on up-to-the-minute sales data.
5. Data Integration: Cloud-based tools typically provide integration with various data sources and applications, making it easier to consolidate sales data from multiple channels, such as online stores, point-of-sale systems, and CRM platforms.
6. Data Security: Reputable cloud providers invest heavily in data security, offering robust security measures, encryption, and compliance with data protection regulations. This can help protect sensitive sales data from cyber threats.
7. Automatic Updates: Cloud-based solutions are regularly updated and maintained by the service provider. This ensures that your sales analysis tools remain current, secure, and functional without the need for manual updates.
8. Collaboration: Cloud platforms often include collaboration features, allowing team members to work together on sales analysis projects in real time. This can streamline decision-making processes and enhance teamwork.
9. Disaster Recovery: Cloud providers typically offer data backup and disaster recovery solutions, reducing the risk of data loss and minimizing downtime in case of unforeseen events.

**Conclusion**

In conclusion, using IBM Cloud for product sales analysis offers businesses a robust and versatile platform to derive insights from their sales data. IBM Cloud provides numerous advantages for organizations looking to enhance their sales analysis capabilities and make more informed decisions. Here are some key points to summarize the benefits of product sales analysis using IBM Cloud:

Scalability: IBM Cloud's flexible infrastructure allows businesses to scale their sales analysis resources as needed, ensuring that they can accommodate growing data volumes and evolving requirements.

Cost-Efficiency: The pay-as-you-go model of IBM Cloud can help businesses manage their budgets effectively, avoiding the upfront costs associated with on-premises hardware and software.

Accessibility: IBM Cloud enables remote access to sales analysis tools and data, facilitating collaboration among team members, even when they are geographically dispersed.

Real-Time Data: The platform supports real-time data processing and analysis, allowing businesses to act on the latest sales information promptly.

Data Integration: IBM Cloud offers robust data integration capabilities, making it easier to consolidate data from multiple sources, including online stores, point-of-sale systems, and customer relationship management (CRM) platforms.

In summary, product sales analysis using IBM Cloud empowers businesses with the tools and infrastructure required to gain deeper insights, optimize decision-making, and effectively leverage their sales data. By combining the advantages of cloud technology with IBM's resources and expertise, businesses can enhance their sales analysis capabilities and ultimately drive better business outcomes. However, organizations should conduct a thorough assessment of their requirements and consider data security and compliance considerations when choosing IBM Cloud or any cloud service provider for sales analysis.