**PHASE 3**

LOADING AND PREPROCESSING THE DATASET

**PRODUCT SALES ANALYSIS**

**INTRODUCTION:**

Product sales analysis is a critical process that businesses undertake to evaluate their sales performance and make informed decisions. It involves examining sales data, trends, and customer behavior to gain insights into what products are selling, to whom, when, and why. By conducting a product sales analysis, businesses can identify their best-selling items, target audience, and market demand, enabling them to optimize their strategies, increase revenue, and enhance customer satisfaction.

1. **Importance of Product Sales Analysis:**

Product sales analysis is crucial for businesses of all sizes, from small startups to large corporations. It serves several key purposes:

1. **Performance Evaluation:**

It helps assess how well specific products are performing in the market. This evaluation includes understanding which products are best-sellers, which are lagging, and why.

1. **Strategic Decision-Making:**

Sales analysis informs strategic decisions, such as product pricing, marketing efforts, inventory management, and expansion plans. Businesses can adjust their strategies based on the data to maximize profits and minimize losses.

**c. Customer Insights:**

By examining sales data, businesses can gain insights into customer preferences, purchase behaviors, and seasonal trends. This information is invaluable for tailoring products and marketing strategies to target audiences effectively.

1. **Inventory Control:**

Effective sales analysis helps in managing inventory more efficiently, reducing carrying costs, and preventing stockouts or overstock situations.

1. **Objectives of Product Sales Analysis:**

The primary objectives of product sales analysis include:

1. **Identifying Trends:**

Recognizing sales trends is essential. This includes understanding which products are consistently popular and which may be losing their appeal.

1. **Assessing Market demand:**

It helps in determining whether a product is meeting or exceeding market demand. Are sales increasing, decreasing, or stagnating?

1. **Comparing Performance:**

Comparing the performance of different products, product categories, or time periods is essential. This enables you to allocate resources effectively.

1. **Profitability Assessment:**

Businesses need to analyze not only revenue but also the profitability of each product. Some products might have high sales but low profit margins, while others may be less popular but more profitable.

1. **Key Metrics and Methods:**

In product sales analysis, various metrics and methods are employed to gain insights into sales data:

**a. Sales Volume:** This metric indicates the total number of products sold within a given time frame. It helps identify the best-sellers.

**b. Sales Revenue:** Sales revenue represents the total monetary value of products sold. It is a critical indicator of a company's financial health.

**c. Profit Margins:** Calculating profit margins is essential to understand how much profit each product contributes. Low-margin products may require re-evaluation.

**d. Customer Segmentation:** Analyzing sales data by customer segments can reveal differences in buying behavior, helping in personalized marketing strategies.

**e. Seasonal Analysis:** Businesses often analyze sales data seasonally to adapt marketing, promotions, and inventory management to the changing demands throughout the year.

**f. Competitive Benchmarking:** Comparing your sales data with competitors' data can provide insights into market share and areas where your products may be underperforming.

**g. Data Visualization:** Using charts, graphs, and dashboards can make complex sales data more understandable and help in spotting trends and anomalies.

**LOADING THE DATASET:**

To load and analyze a dataset for product sales using machine learning, follow these steps in Python:

1. Import Required Libraries:

Import libraries for data manipulation, visualization, and machine learning.

Python:

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

pd.options.display.max\_columns=50

sns.set(style="darkgrid")

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

from sklearn.linear\_model import LinearRegression

2. Read the Dataset:

Load your product sales dataset using Pandas.

Python:

data = pd.read\_csv('/kaggle/input/product-sales-data/statsfinal.csv')

data.head(-1)

Replace 'product\_sales\_data.csv' with the actual file path to your dataset.

# 

3. Data Preprocessing:

Prepare your dataset for analysis. This may involve handling missing data, encoding categorical variables, and scaling/normalizing features, depending on your specific dataset.

# Check for missing values

In [6]:

data.isnull().sum()

Out[6]:

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

**Observations:**

* we have no missing data

**Note:**

* No missing values in a dataset is not common.
* while working with fresh data, you will have to do a ton of cleaning, this will result in some missing or lost data.
* Look into "feature engineering" and "missing value handling" for ways to resolve this issues.

4. Split the Dataset:

Divide your dataset into features (X) and the target variable (y), and then split it into a training and testing set.

Python:

X = df.drop('sales', axis=1) # Replace 'sales' with your target column

y = df['sales']

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

# 

**Observations:**

* The train dataset has 4600 entries(rows) and 9 columns. (we dropped one column)
* Date is an object data type. the rest of numerical in nature.

5. Choose a Machine Learning Model:

Select an appropriate machine learning model for sales prediction, such as Linear Regression.

python

model = LinearRegression()

6. Train the Model:

Fit your machine learning model to the training data.

python

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

# Checking the info of the training data:

In [5]:

data.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 4600 entries, 0 to 4599

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Date 4600 non-null object

1 Q-P1 4600 non-null int64

2 Q-P2 4600 non-null int64

3 Q-P3 4600 non-null int64

4 Q-P4 4600 non-null int64

5 S-P1 4600 non-null float64

6 S-P2 4600 non-null float64

7 S-P3 4600 non-null float64

8 S-P4 4600 non-null float64

dtypes: float64(4), int64(4), object(1)

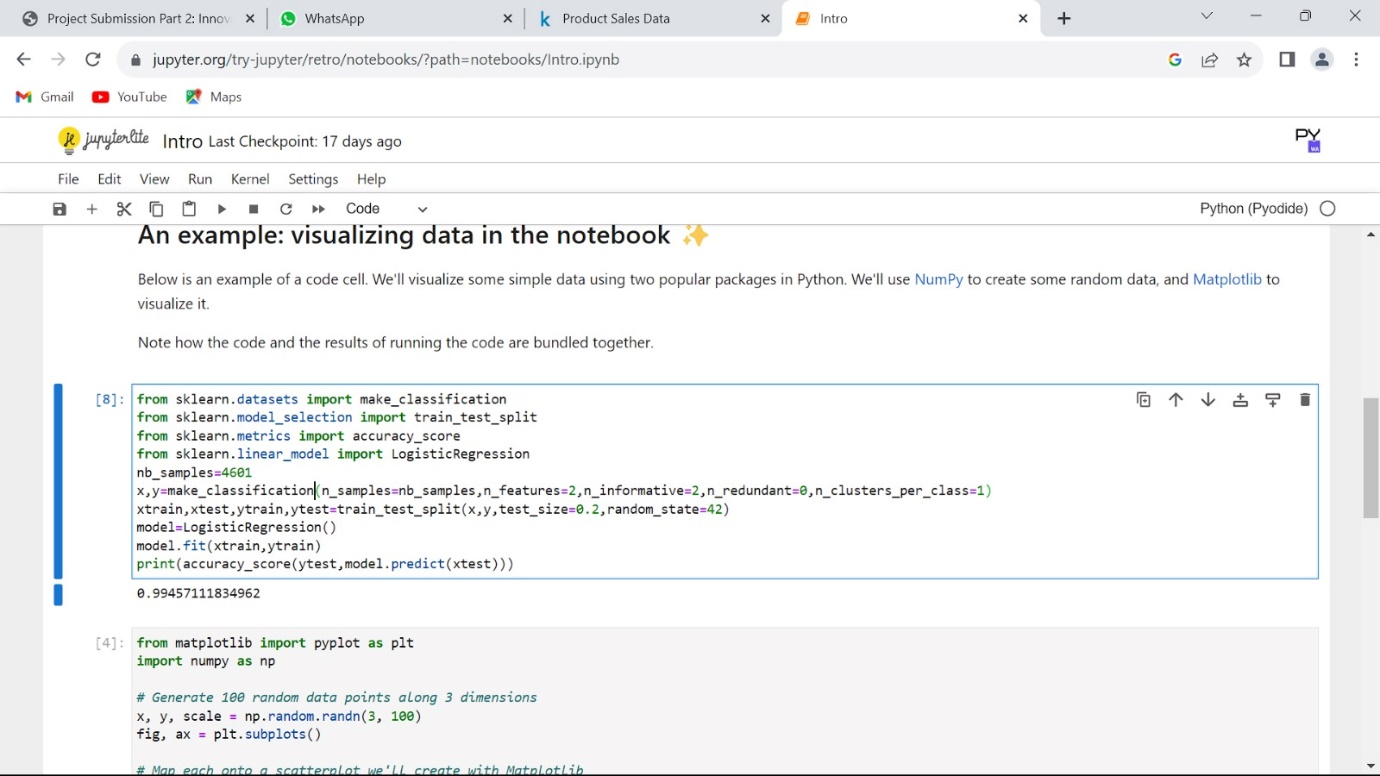
memory usage: 323.6+ KB

linkcode

**Observations:**

* The train dataset has 4600 entries(rows) and 9 columns. (we dropped one column)
* Date is an object data type. the rest of numerical in nature.

**ACCURACY:**

****

EXPLORATORY DATA ANALYSIS:

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));

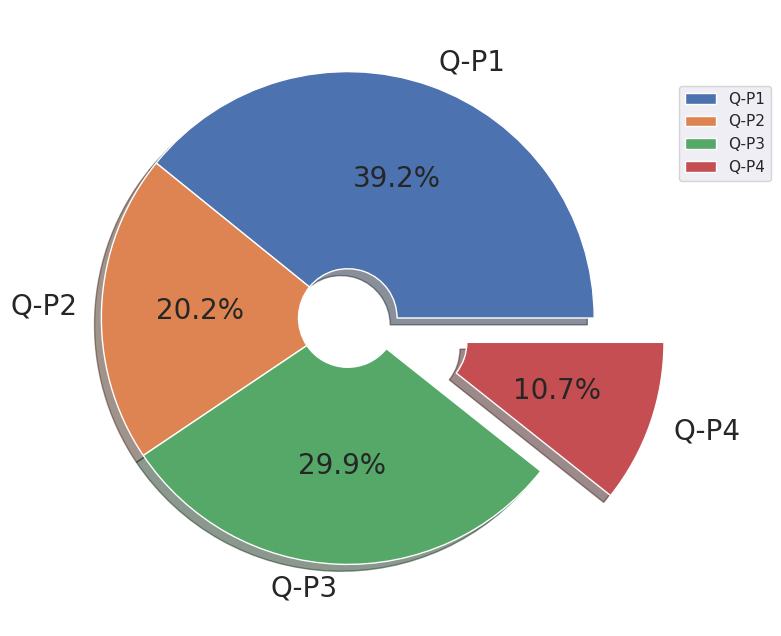
Q-P1 18960506

Q-P2 9799295

Q-P3 14470404

Q-P4 5168100

dtype: int6



*# which is the most occuring month*

print(df["month"].value\_counts())

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

sns.countplot(x="month",data=df,edgecolor="black")

plt.xticks(rotation=90);

October 411

January 399

July 398

June 385

August 385

September 385

November 385

December 385

March 380

May 379

April 367

February 341

Name: month, dtype: int64

sns.relplot(x="month",y="S-P1",data=df,kind="line",height=10,color="red")

plt.xticks(rotation=90);

sns.relplot(x="month",y="S-P2",data=df,kind="line",height=10,color="blue")

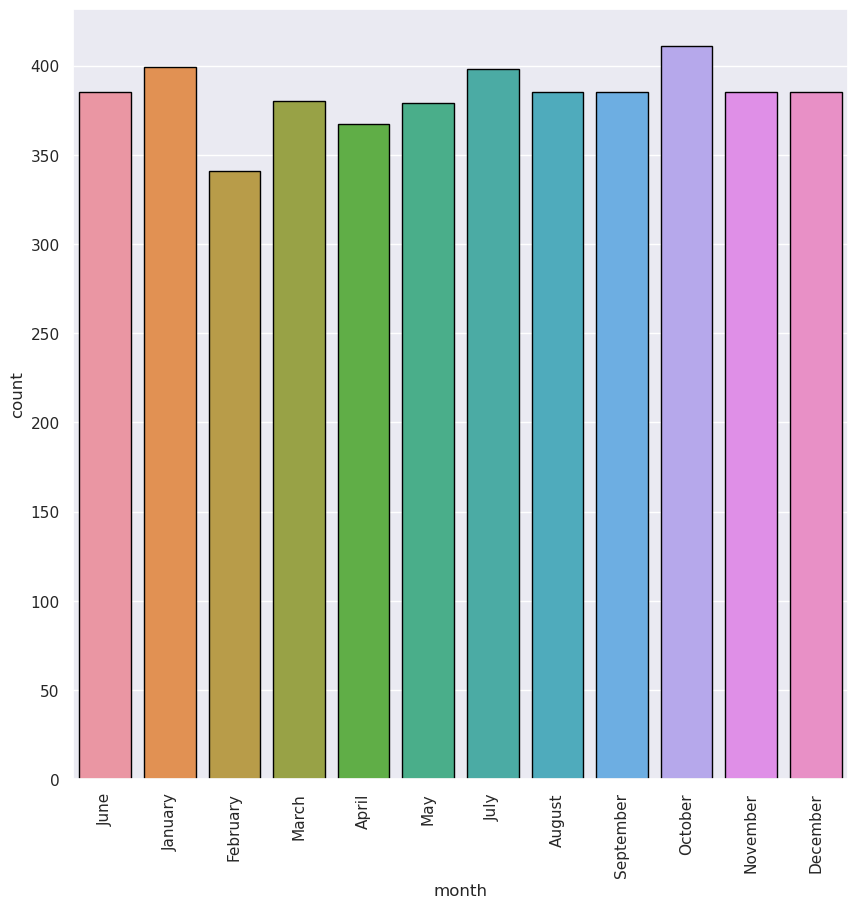
plt.xticks(rotation=90);

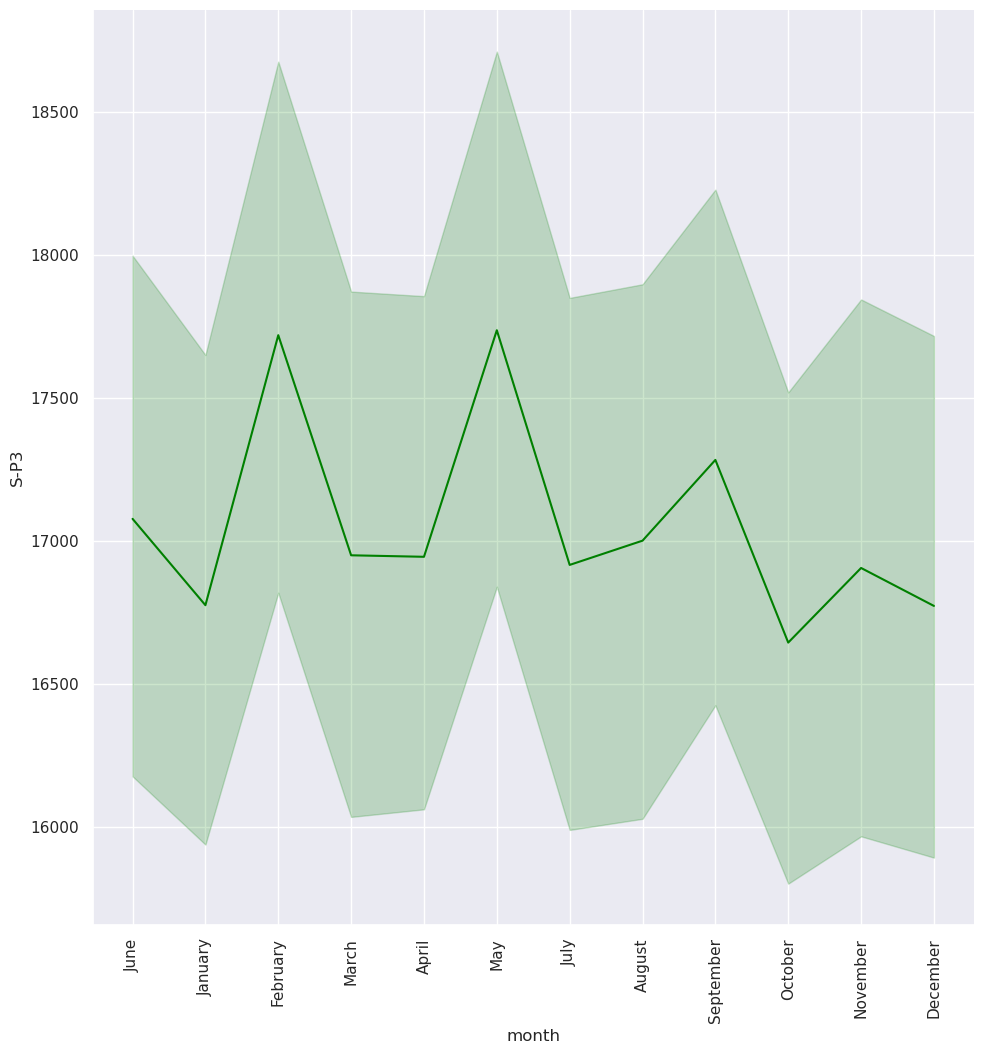
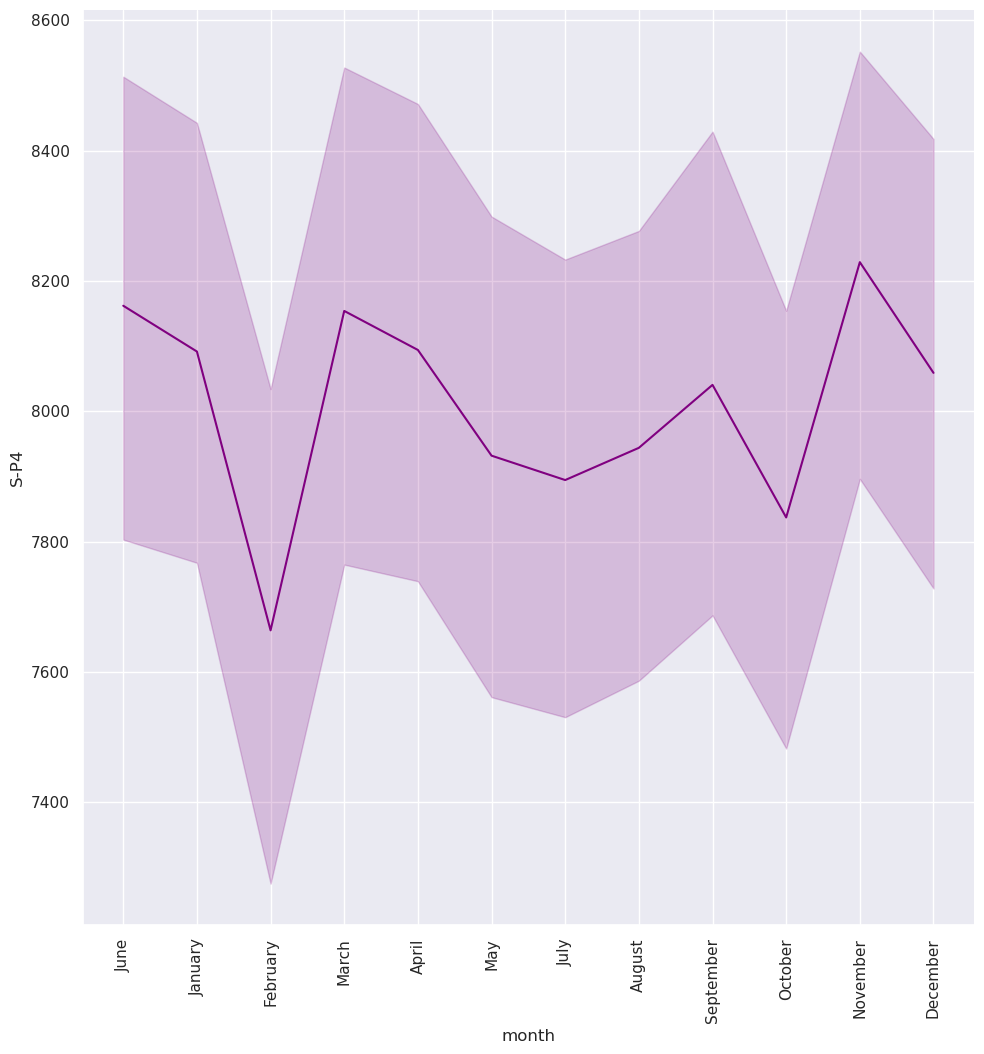
sns.relplot(x="month",y="S-P3",data=df,kind="line",height=10,color="green")

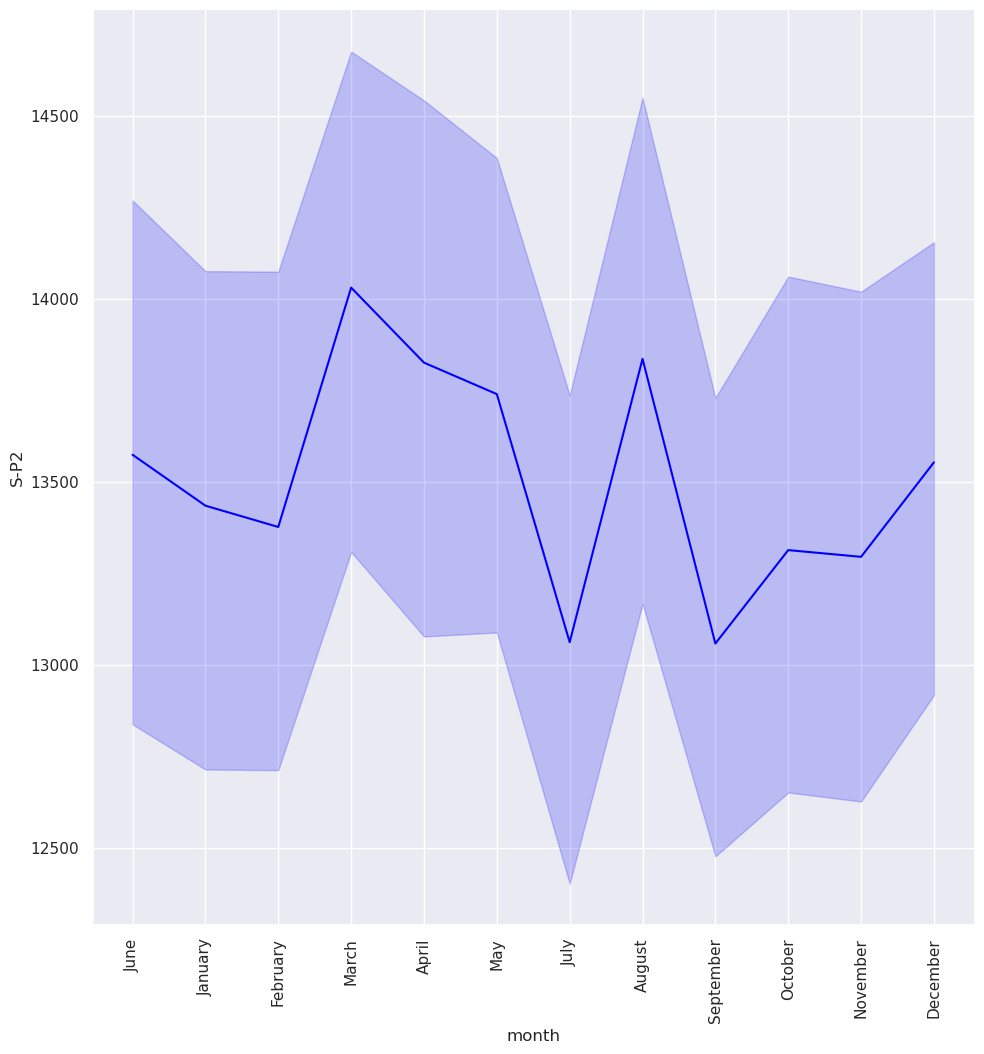
plt.xticks(rotation=90);

sns.relplot(x="month",y="S-P4",data=df,kind="line",height=10,color="purple")

plt.xticks(rotation=90);







 **CONCLUSION**:

In conclusion, loading and preprocessing data are foundational steps in product sales analysis that significantly impact the accuracy and reliability of insights derived from the data. Loading data involves gathering information from various sources and organizing it into a format suitable for analysis, ensuring that no vital details are omitted. On the other hand, data preprocessing involves cleaning, transforming, and enriching the data to enhance its quality and usability. These preparatory steps are crucial for removing inconsistencies, handling missing values, and standardizing formats, enabling the data to be effectively analyzed using statistical and machine learning techniques.

A well-executed data loading and preprocessing phase ensures that the subsequent analysis is based on reliable, consistent, and comprehensive data. It also sets the stage for uncovering valuable patterns, trends, and correlations in product sales, which can inform strategic business decisions. Therefore, investing time and effort into careful data loading and preprocessing is indispensable for deriving meaningful insights and driving informed actions to optimize product sales and overall business performance.