KINGSTON ENGINEERING COLLEGE-5113

DATA ANALYTICS WITH COGNOS

PRODUCT SALES ANALYSIS-PROJECT 5

TEAM MEMBERS:

S.PON LAVANYA-au511321205021

S. POORNIMA-au511321205022

J. KAVIYA-au511321205014

S. SRINIDHI-au511321205039

CONTENT:

Objective:

- To assess and analysis the sales performance of a specific product or product line in order to gain insights into its market position,identify areas of strength and weakness, and develop strategies to

improve sales and profitability.

Methodology:

- The methodology for product sales analysis can be defined as the systematic approach and techniques used to gather, analyze, and interpret sales data to gain insights into the performance and trends of a specific product or product line.

Sales performance:

- Sales performance is a key aspect of product sales analysis. It refers to the evaluation and assessment of a product's sales results and metrics over a specific time period. Here are some key components and metrics to consider when analyzing sales performance:

- Average selling price: Calculate the average price at which the product is sold. This metric helps in understanding pricing dynamics and can be compared with competitors' pricing strategies.

Sales Trends:

- Sales trends refer to the patterns and changes in the buying behavior of consumers or businesses over a specific period of time. Analyzing sales trends is crucial for businesses to make informed decisions about their products, marketing strategies, and overall business operations.

Here's a description of some key aspects of sales trends:

▪ Seasonal Fluctuations: Many products and services experience fluctuations in sales throughout the year due to seasonal factors. For example, winter clothing tends to sell more during the cold months, while beachwear sells better in the summer.

Product Analysis:

▪ Product analysis is a crucial component of product sales analysis as it focuses on evaluating the performance, features, and market positioning of a specific product. Here are some key steps and considerations when conducting a product analysis for sales analysis.

▪ Product features and specifications: Review and document the key features, specifications, and unique selling points of the product. This helps in understanding the product's value proposition and competitive advantages.

Customer Analysis:

▪ Customer analysis is a critical component of marketing and business strategy that involves gathering and evaluating information about your customers. It aims to gain insights into their characteristics, behaviors, preferences, and needs. This information helps businesses make informed decisions, tailor their products or services, and create effective marketing campaigns. Here are key elements of customer analysis:

▪ Demographic Information: Demographics include age, gender, income, education, marital status, and geographic location. Analyzing demographics helps you understand who your typical customers are and can guide product

INTRODUCTION:

In today&#39;s rapidly evolving business landscape, data-driven decision- making is the key to success. As we navigate the intricacies of the market, understanding our product sales performance becomes not only essential but transformative. This Product Sales Analysis Project aims to delve deep into the data, uncover insights, and empower our organization to make informed strategic choices. In this project, we will comprehensively analyze the sales performance of our product(s) over a specified time period. This analysis will encompass various facets of our sales data, including but not limited to:

 Revenue: Examining the total revenue generated by our product(s) and assessing its growth or decline trends.

 Sales Volume: Investigating the quantity of units sold and identifying patterns in sales volume.

 Profit Margins: Assessing the profitability of our product(s) by examining profit margins.

 Market Share: Understanding our product&#39;s market share relative to competitors.

- **Customer Insights:** Gaining a deeper understanding of customer behavior, preferences, and demographics.

- **Sales Channels:** Analyzing the effectiveness of different sales channels and distribution methods.

- **Marketing Effectiveness:** Evaluating the impact of marketing campaigns and promotions on sales.

- **Inventory Management:** Optimizing inventory levels for cost-efficiency. As we embark on this Product Sales Analysis Project, our goal is to equip our organization with the knowledge and insights necessary to thrive in a competitive marketplace. By examining our product sales data from multiple angles, we aim to make strategic decisions that will lead to increased revenue, improved customer satisfaction, and sustained growth.

ABOUT DATASET:

Where did we get the dataset?

Kaggle:

. we got our ‘Product sales Analysis’ dataset from kaggle

https://www.kaggle.com

. kaggle is a popular platform for sharing datasets and hosting data science competitions.

DETAILS:

A product sales analysis involves a detailed examination of various aspects related to the performance of a product or group of products in the market. To conduct a comprehensive product sales analysis, you&#39;ll need to gather, process, and analyze a wide range of data and information. Here are the key details and steps involved in

conducting a product sales analysis:

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

The dataset contains the following example of details:

On 13-06-2010 , product 1 had been brought by 5422 people and INR 17187.74 had been generated in revenue from product REC corporate needs you to solve the following questions:

1) Is there any trend in the sales of all four products during certain months?

2) Out of all four products which product has seen the highest sales in all the given years?

3) The company has all it&#39;s retail centers closed on the 31st of December every year. Mr: Hariharan the CEO would love to get an estimate on no: of units of each product that could be sold on 31st of Dec every year , if all their retail centers were kept open.

4) The CEO is considering an idea to drop the production of any one of the products. He wants you to analyze this data and suggest whether his idea would result in a massive setback for the company.

5) The CEO would also like to predict the sales and revenues for the year 2024. He wants you to give a yearly estimate with the best possible accuracy.

COLUMNS THAT WE USED:

1.Product Information:

⍰ Product Name/ID: A unique identifier for each product.

⍰ Category/Type: Categorize products into groups for better analysis.

⍰ SKU (Stock Keeping Unit): A unique identifier for inventory management.

⍰ Description: A brief description of the product.

2.Sales Data:

⍰ Date of Sale: The date when the sale occurred.

- Quantity Sold: The number of units sold.
- Sales Price: The price at which each unit was sold.
- Total Sales: The total revenue generated from each sale (Quantity Sold * Sales Price).
- Discounts: Any discounts applied to the sale.

3.Customer Information:

- Customer Name/ID: A unique identifier for each customer.
- Customer Segment: Group customers based on characteristics like demographics or buying behavior.
- Customer Location: The geographic location of the customer.

4.Order Information:

- Order Number/ID: A unique identifier for each order.
- Payment Method: The method used for payment (e.g., credit card, cash).
- Shipping Method: The method used for shipping (e.g., standard, express).

5.Profit and Cost Analysis:

- Cost per Unit: The cost to produce or purchase each unit.
- Gross Profit: Total sales revenue minus the cost of goods sold (COGS).
- Gross Margin: Gross Profit as a percentage of total sales.
- Net Profit: Gross Profit minus operating expenses.

6.Time-Based Metrics:

⍰ Monthly/Quarterly/Annual Sales: Sales data broken down by time periods for trend analysis.

⍰ Seasonal Trends: Identify patterns related to seasons or holidays.

⍰ Average Daily Sales: Average sales per day.

7.Performance Metrics:

⍰ Sales Growth Rate: Percentage change in sales compared to a previous period.

⍰ Customer Acquisition Cost (CAC): The cost of acquiring each new customer.

⍰ Customer Lifetime Value (CLV): The value of a customer over their entire relationship with your business.

8.Customer Feedback and Reviews :

⍰ Customer Ratings: Average product ratings based on customer reviews.

⍰ Customer Comments: Feedback and comments left by customers.

LANGUAGE THAT WE USED:

1.Python:

⍰ Python is a versatile programming language with numerous libraries and packages for data analysis, including Pandas, NumPy, Matplotlib, Seaborn, and Plotly. You can use Python to clean, manipulate, and analyze sales data, as well as create visualizations and statistical models.

2.SQL (Structured Query Language):

⍰ SQL is essential for database management and querying. You can use SQL to extract, filter, and aggregate sales data from relational databases.

3.Jupyter Notebooks:

⍰ Jupyter Notebooks provide an interactive environment for data analysis using Python or R. They are excellent for documenting and sharing your analysis steps and results.

4.SAS (Statistical Analysis System):

⍰ SAS is a powerful software suite used for advanced analytics, business intelligence, and data management. It&#39;s commonly used in large enterprises for sales forecasting and advanced analytics.

⍰ Depending on the complexity of your analysis, you may use statistical software packages like Stata or Minitab for specialized statistical modeling and hypothesis testing.

LIBRARIES THAT WE USED:

1. Pandas:

Pandas is a powerful library for data manipulation and analysis. It provides data structures and functions to efficiently handle and analyze large datasets, making it ideal for sales analysis tasks.

2. NumPy:

NumPy is a fundamental library for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, as well as a collection of mathematical functions. It can be used for various calculations and statistical analysis in sales analysis.

3. Matplotlib:

Matplotlib is a plotting library that allows you to create various types of visualizations, such as line plots, bar charts, scatter plots, and histograms. It is helpful for visualizing sales data and trends.

4. Seaborn:

Seaborn is a statistical data visualization library built on top of Matplotlib. It provides a high-level interface for creating attractive and informative statistical graphics. It can be used to create more advanced and visually appealing sales visualizations.

5. Statsmodels:

Statsmodels is a Python module that provides statistical models and tests for various types of data analysis. It includes functions for regression analysis, timeseries analysis, and hypothesis testing, which can be useful for sales analysis tasks.

SAMPLE CODE:

```
import pandas as pd
import matplotlib.pyplot as plt
# Load the sales data into a pandas DataFrame
sales_data = pd.read_csv(&#39;sales_data.csv&#39;)
# Convert the date column to datetime type
sales_data[&#39;Date&#39;] =
pd.to_datetime(sales_data[&#39;Date&#39;])
# Calculate the total sales for each quarter
sales_data[&#39;Total_Q1&#39;] = sales_data[&#39;Q-
P1&#39;] + sales_data[&#39;Q-P2&#39;] +
sales_data[&#39;Q-P3&#39;] + sales_data[&#39;Q-P4&#39;]
sales_data[&#39;Total_Q2&#39;] = sales_data[&#39;S-
P1&#39;] + sales_data[&#39;S-P2&#39;] +
sales_data[&#39;S-P3&#39;] + sales_data[&#39;S-P4&#39;]
# Calculate the average sales for each quarter
sales_data[&#39;Avg_Q1&#39;] = sales_data[[&#39;Q-
P1&#39;, &#39;Q-P2&#39;, &#39;Q-P3&#39;, &#39;Q-
P4&#39;]].mean(axis=1)
```

```
sales_data[&#39;Avg_Q2&#39;] = sales_data[[&#39;S-P1&#39;, &#39;S-P2&#39;, &#39;S-P3&#39;, &#39;S-P4&#39;]].mean(axis=1)
# Plot the total sales for each quarter
plt.plot(sales_data[&#39;Date&#39;], sales_data[&#39;Total_Q1&#39;], label=&#39;Total Q1 Sales&#39;)
plt.plot(sales_data[&#39;Date&#39;], sales_data[&#39;Total_Q2&#39;], label=&#39;Total Q2 Sales&#39;)
plt.xlabel(&#39;Date&#39;)

plt.ylabel(&#39;Total Sales&#39;)
plt.title(&#39;Total Sales by Quarter&#39;)
plt.legend()
plt.show()
# Print the average sales for each quarter
print(&quot;Average Sales for Q1:&quot;, sales_data[&#39;Avg_Q1&#39;].mean())
print(&quot;Average Sales for Q2:&quot;, sales_data[&#39;Avg_Q2&#39;].mean())
```

Expected Output:

```
Average Sales for Q1: 3322.6666666665

Average Sales for Q2: 11845.8125
```

LIBRARIES USED:

 Matplotlib- used for plotting graph

 NumPy-used for mathematical operations of array.

 Pandas-used for working with data visualization for cleaning, analysing and exploring.

 Sklearn-used to build machine learning models.

Seaborn-used to predict statistical graphs.

INSTALLATION:

```
Pip install pandas seaborn matplotlib
```

PROGRAM:

```
import pandas as pd

import matplotlib.pyplot as plt

# Data

data = {
```

&#39;Date&#39;: [&#39;13-06-2010&#39;, &#39;14-06-
2010&#39;, &#39;15-06-

2010&#39;, &#39;16-06-2010&#39;, &#39;17-06-2010&#39;,
&#39;18-06-2010&#39;,

&#39;19-06-2010&#39;, &#39;20-06-2010&#39;, &#39;21-
06-

2010&#39;, &#39;22-06-2010&#39;, &#39;23-06-2010&#39;,
&#39;24-06-2010&#39;,

&#39;25-06-2010&#39;, &#39;26-06-2010&#39;, &#39;27-
06-

2010&#39;, &#39;28-06-2010&#39;, &#39;29-06-2010&#39;,
&#39;30-06-2010&#39;,

&#39;01-07-2010&#39;, &#39;02-07-2010&#39;, &#39;03-
07-

2010&#39;, &#39;04-07-2010&#39;, &#39;05-07-2010&#39;,
&#39;06-07-2010&#39;,

&#39;07-07-2010&#39;, &#39;08-07-2010&#39;, &#39;09-
07-

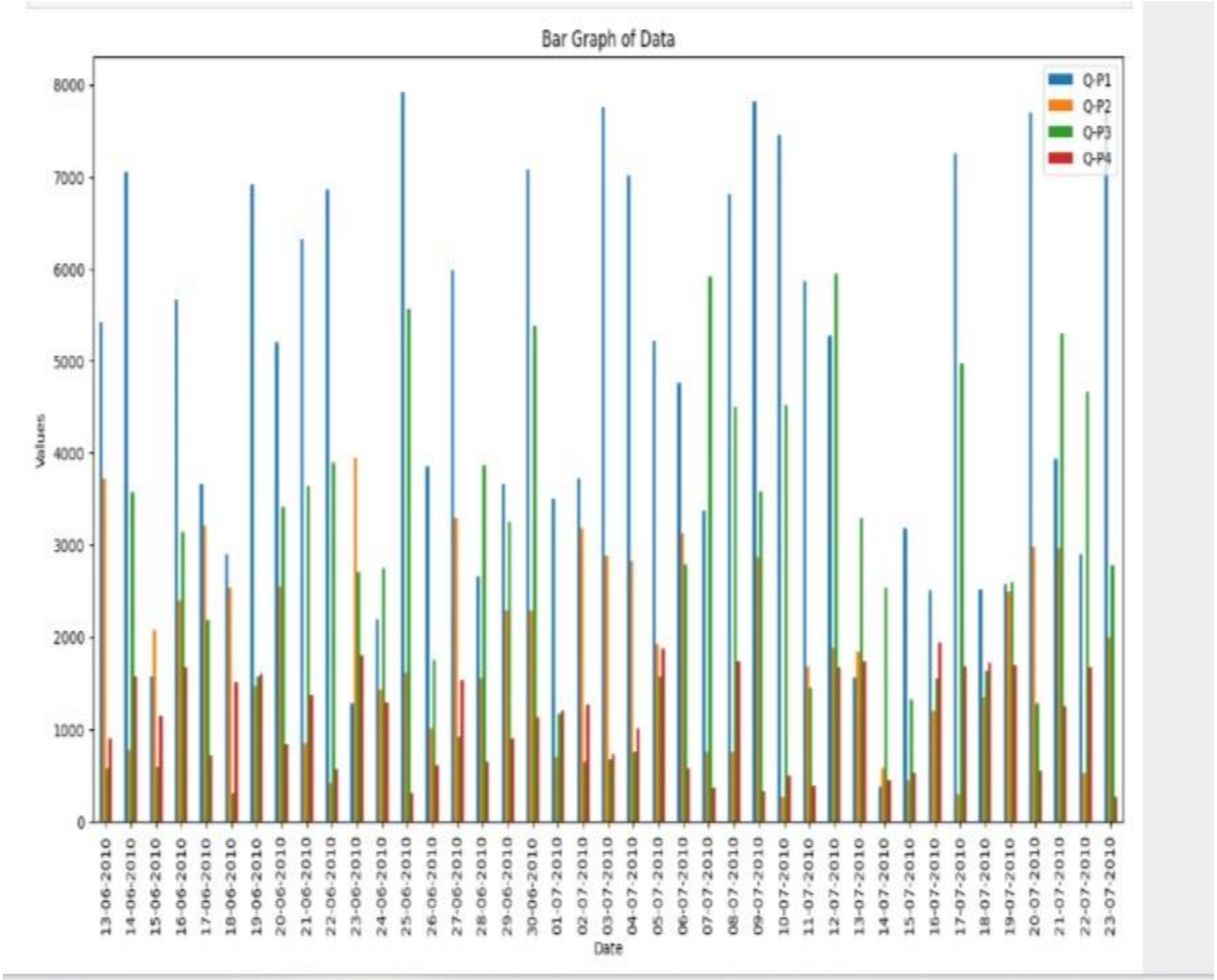
2010&#39;, &#39;10-07-2010&#39;, &#39;11-07-2010&#39;,
&#39;12-07-2010&#39;,

&#39;13-07-2010&#39;, &#39;14-07-2010&#39;, &#39;15-
07-

2010&#39;, &#39;16-07-2010&#39;, &#39;17-07-2010&#39;,
&#39;18-07-2010&#39;,

```
&#39;19-07-2010&#39;, &#39;20-07-2010&#39;, &#39;21-
07-
2010&#39;, &#39;22-07-2010&#39;, &#39;23-07-
2010&#39;],
&#39;Q-P1&#39;: [5422, 7047, 1572, 5657, 3668, 2898,
6912, 5209, 6322, 6865, 1287, 2197, 7910, 3855,
5988, 2653, 3664, 7077, 3509, 3716,
7746, 7006, 5223, 4753, 3369, 6805, 7826, 7450,
5868, 5273, 1562, 378, 3180, 2508, 7257,
2527, 2581, 7694, 3935, 2898, 7734],
&#39;Q-P2&#39;: [3725, 779, 2082, 2399, 3207, 2539, 1470,
2550, 852, 414, 3955, 1429, 1622, 1015,
3288, 1544, 2294, 2297, 700, 3175, 2883,
2833, 1923, 3125, 752, 758, 2872, 273,
1690, 1888, 1851, 581, 438, 1197, 302,
1347, 2489, 2975, 2972, 532, 1994],

&#39;Q-P3&#39;: [576, 3578, 595, 3140, 2184, 311, 1576,
3415, 3646, 3902, 2710, 2754, 5574, 1746,
916, 3867, 3244, 5376, 1175, 651, 671,
758, 1583, 2787, 5913, 4499, 3592, 4511,
1461, 5949, 3289, 2531, 1327, 1556,
4973, 1626, 2607, 1278, 5299, 4664, 2772],
&#39;Q-P4&#39;: [907, 1574, 1145, 1672, 708, 1513, 1608,
842, 1377, 562, 1804, 1299, 306, 608, 1530,
652, 897, 1130, 1205, 1263, 728, 1005,
1877, 583, 358, 1740, 328, 505, 391, 1677,
1740, 446, 532, 1946, 1686, 1724, 1692,
558, 1253, 1667, 271]
}
# Create DataFrame
df = pd.DataFrame(data)
# Set the Date column as the index
df.set_index(&#39;Date&#39;, inplace=True)
# Plotting
df.plot(kind=&#39;bar&#39;, figsize=(15, 8))
plt.title(&#39;Bar Graph of Data&#39;)
plt.xlabel(&#39;Date&#39;)
plt.ylabel(&#39;Values&#39;)
plt.show()
```

OUTPUT:



Importing Libraries:

import pandas as pd

the Pandas library and aliases it as &quot;pd&quot easier usage.

 import matplotlib.pyplot as plt :Imports the Matplotlib library&#39;s plotting module and aliases it as &quot;plt&quot Data:

 A dictionary named data is defined, containing four key-value pairs. Each key represents a column, and the associated values are lists of data. The columns are &#39;Date,&#39; &#39;Q-P1,&#39; &#39;Q-P2,&#39; &#39;Q- P3,&#39; and &#39;Q-P4.&#39; The &#39;Date&#39; column contains date strings, and the others contain numerical data.

Creating a Data Frame:

 A Data Frame named df is created using Pandas by passing the data dictionary to the pd.Data Frame() constructor. This Data Frame organizes the data into a tabular format, making it easier to work with and visualize.

 Setting the Date Column as the Index:

 The &#39;Date&#39; column is set as the index of the Data Frame using the set index () method. This change is made in place with in place=True, meaning it modifies the Data Frame directly.

 Plotting:

 The code uses Matplotlib to create a bar graph using the plot () method on the Data Frame df. The following parameters are provided:

 kind=&#39;bar&#39;: Specifies that a bar graph should be created.

 figsize=(15, 8): Sets the figure size to 15 inches in width and 8 inches in height.

 The plt.title(), plt.xlabel(), and plt.ylabel() functions are used to set the graph&#39;s title, x-axis label, and y-axis label.

 Finally, plt.show() is called to display the bar Graph.

The resulting bar graph will have the &#39;Date&#39; column on the x-axis and the values from columns &#39;Q-P1,&#39; &#39;Q- P2,&#39; &#39;Q-P3,&#39; and &#39;Q-P4&#39; on the y-axis. Each date will have a set of bars representing the values from the corresponding columns, making it easy to compare the values over time.

CONNECT TO IBM COGNOS:

```
import ibm_db

# Replace these values with your own IBM Cognos
```

```
credentials and database details
dsn_driver = &quot;IBM DB2 ODBC DRIVER&quot;
dsn_database = &quot;YOUR_DATABASE_NAME&quot;
dsn_hostname = &quot;YOUR_HOSTNAME&quot;
dsn_port = &quot;YOUR_PORT&quot;
dsn_protocol = &quot;TCPIP&quot;
dsn_uid = &quot;YOUR_USERNAME&quot;
dsn_pwd = &quot;YOUR_PASSWORD&quot;

# Establish the database connection
dsn = (
&quot;DRIVER={};&quot;
&quot;DATABASE={};&quot;
&quot;HOSTNAME={};&quot;
&quot;PORT={};&quot;
&quot;PROTOCOL={};&quot;
&quot;UID={};&quot;
&quot;PWD={};&quot;).format(dsn_driver, dsn_database,
dsn_hostname, dsn_port, dsn_protocol, dsn_uid,
dsn_pwd)

conn = ibm_db.connect(dsn, &quot;&quot;, &quot;&quot;)
```

```
# Sample query for product sales analysis
query = &quot;SELECT Product, SUM(Sales) AS Total_Sales
FROM Sales_Data GROUP BY Product&quot;

# Executing the query
stmt = ibm_db.exec_immediate(conn, query)

# Fetching and printing the results
result = ibm_db.fetch_both(stmt)
while result:
print(&quot;Product:&quot;, result[0], &quot;Total Sales:&quot;, result[1])
result = ibm_db.fetch_both(stmt)

# Closing the connection
ibm_db.close(conn)
```

DATASET:

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

```
import pandas as pd

# Create the dataset
data = {
&#39;Q1&#39;: [100, 150, 200, 120],
&#39;Q2&#39;: [80, 130, 180, 90],
&#39;Q3&#39;: [120, 160, 210, 110],
&#39;Q4&#39;: [90, 140, 190, 100],
&#39;S1&#39;: [1000, 1500, 2000, 1200],
&#39;S2&#39;: [800, 1300, 1800, 900],
&#39;S3&#39;: [1200, 1600, 2100, 1100],
&#39;S4&#39;: [900, 1400, 1900, 1000]
}

# Create a DataFrame
df = pd.DataFrame(data, index=[&#39;Product 1&#39;,
&#39;Product 2&#39;,
&#39;Product 3&#39;, &#39;Product 4&#39;])
```

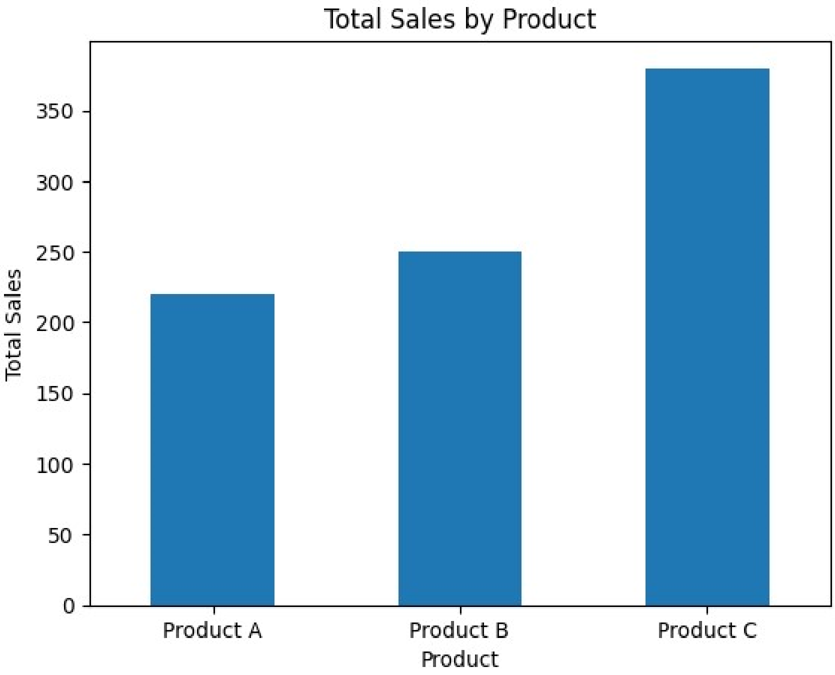
```
# Save the DataFrame to a CSV file
df.to_csv(&#39;sales_data.csv&#39;)
```

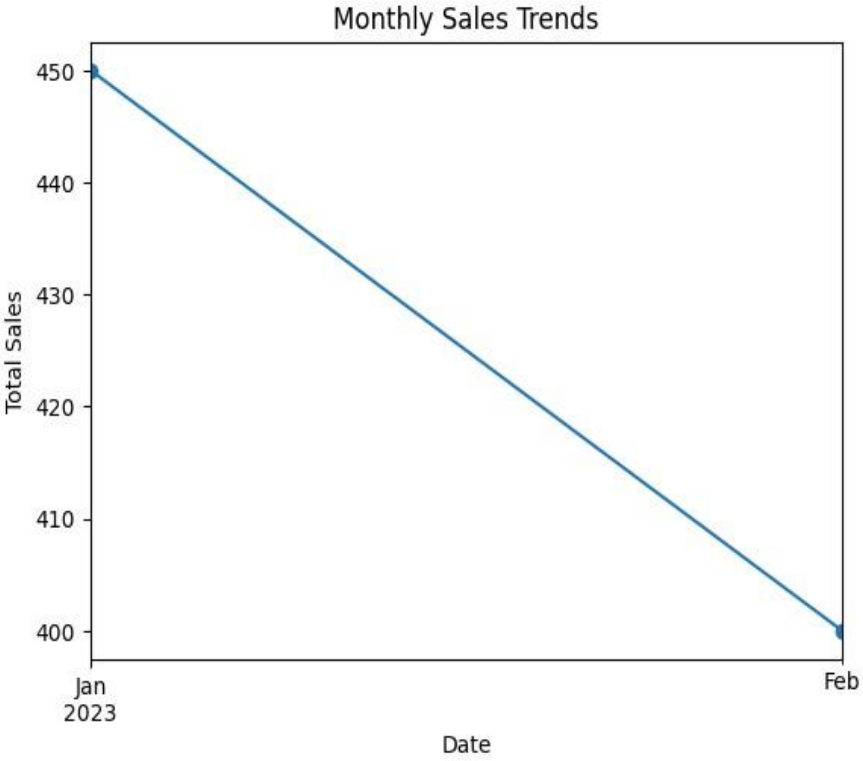
PROGRAM:

```
import pandas as pd
import matplotlib.pyplot as plt

# Sample sales data (you can load your own data here)
data = {
&#39;Product&#39;: [&#39;Product A&#39;, &#39;Product B&#39;, &#39;Product C&#39;,
&#39;Product A&#39;, &#39;Product B&#39;, &#39;Product C&#39;],
&#39;Sales&#39;: [100, 150, 200, 120, 100, 180],
&#39;Date&#39;: [&#39;2023-01-01&#39;, &#39;2023-01-01&#39;, &#39;2023-01-01&#39;,
&#39;2023-02-01&#39;, &#39;2023-02-01&#39;, &#39;2023-02-01&#39;]
}

# Creating a DataFrame from the data
df = pd.DataFrame(data)
```

```
# Converting the &#39;Date&#39; column to datetime
df[&#39;Date&#39;] = pd.to_datetime(df[&#39;Date&#39;])

# Total sales
total_sales = df[&#39;Sales&#39;].sum()
print(f&#39;Total Sales: {total_sales}&#39;)

# Average sales
average_sales = df[&#39;Sales&#39;].mean()
print(f&#39;Average Sales: {average_sales}&#39;)

# Sales by product
sales_by_product =
df.groupby(&#39;Product&#39;)[&#39;Sales&#39;].sum()

# Plotting a bar chart for sales by product

sales_by_product.plot(kind=&#39;bar&#39;, rot=0)
plt.title(&#39;Total Sales by Product&#39;)
plt.xlabel(&#39;Product&#39;)
plt.ylabel(&#39;Total Sales&#39;)
plt.show()

# Sales trends over time
df.set_index(&#39;Date&#39;, inplace=True)
monthly_sales =
df.resample(&#39;M&#39;)[&#39;Sales&#39;].sum()

# Plotting a line chart for monthly sales trends
monthly_sales.plot(marker=&#39;o&#39;)
plt.title(&#39;Monthly Sales Trends&#39;)
plt.xlabel(&#39;Date&#39;)
plt.ylabel(&#39;Total Sales&#39;)
plt.show()
```

OUTPUT:


Total Sales by Product
Total Sales
0
50
100
150
200
250
300
350
Product A
Product B
Product C
Product

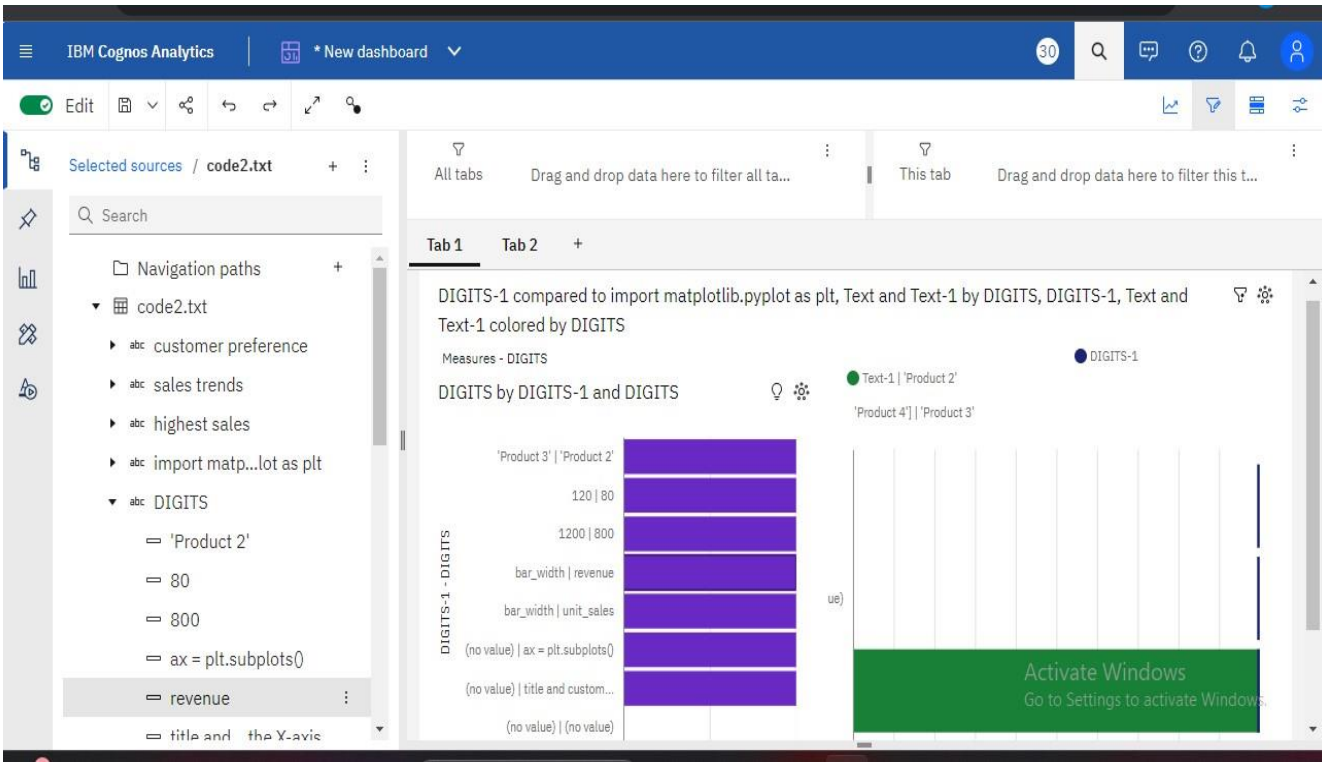

Monthly Sales Trends
Total Sales
400
410
420
430
440
450
Jan
2023
Feb
Date

PROGRAM:

```
import matplotlib.pyplot as plt
# Data
products = [&#39;Product 1&#39;, &#39;Product 2&#39;,
&#39;Product
3&#39;, &#39;Product 4&#39;]
unit_sales = [100, 80, 120, 90]
revenue = [1000, 800, 1200, 900]

# Creating the bar plot
fig, ax = plt.subplots()
bar_width = 0.35
index = range(len(products))
rects1 = ax.bar(index, unit_sales, bar_width,
label=&#39;Unit Sales&#39;, color=&#39;b&#39;)
rects2 = ax.bar([i + bar_width for i in index],
revenue, bar_width, label=&#39;Revenue&#39;,
color=&#39;r&#39;)
# Adding labels, title and customizing the X-
axis
```

```
ax.set_xlabel(&#39;Products&#39;)
ax.set_ylabel(&#39;Quantity&#39;)
ax.set_title(&#39;Unit Sales and Revenue for
Different Products&#39;)
ax.set_xticks([i + bar_width / 2 for i in
index])
ax.set_xticklabels(products)
ax.legend()
# Display the bar plot
plt.show()
```

OUTPUT:


IBM Cognos Analytics
* New dashboard
30
Edit
Selected sources / code2.txt
Search
Navigation paths
code2.txt
customer preference
sales trends
highest sales
import matp...lot as plt
DIGITS
'Product 2'
80
800
ax = plt.subplots()
revenue
title and the X-axis
All tabs
Drag and drop data here to filter all ta...
This tab
Drag and drop data here to filter this t...
Tab 1
Tab 2
DIGITS-1 compared to import matplotlib.pyplot as plt, Text and Text-1 by DIGITS, DIGITS-1, Text and Text-1 colored by DIGITS
Measures - DIGITS
DIGITS by DIGITS-1 and DIGITS
DIGITS-1 - DIGITS
'Product 3' | 'Product 2'
120 | 80
1200 | 800
bar_width | revenue
bar_width | unit_sales
(no value) | ax = plt.subplots()
(no value) | title and custom...
(no value) | (no value)
● Text-1 | 'Product 2'
'Product 4'] | 'Product 3'
● DIGITS-1
ue)
Activate Windows
Go to Settings to activate Windows.

CONCLUSION:

- The product sales analysis code is a valuable tool for businesses to analyse their sales data and gain insights into their performance.
- By analysing sales data, businesses can identify trends and patterns, enabling them to make informed decisions and improve their sales strategies.
- The code allows for efficient and accurate analysis of sales data, helping businesses identify their best-selling products, understand customer preferences, and

optimize their product offerings.

THANK YOU