**PRODUCT SALES ANALYSIS**

**PHASE-2**

**NAME: NAGENTHIRA KUMAR**

**NM\_ID: au211521106104**

**INTRODUCTION:**

**\*Strategic Decision-Making:** Product sales analysis is a pivotal tool for businesses to make data-driven decisions. It empowers companies to assess their product portfolio, optimize marketing strategies, and improve operational efficiency.

**\*Market Insights and Trends:** This analysis helps in understanding market trends and customer preferences, enabling companies to anticipate demands, align product development, and maintain a competitive edge.

**\*Enhanced Profitability:** By harnessing the power of data and analytics, product sales analysis aids in allocating resources effectively, enhancing customer satisfaction, and ultimately increasing profitability, making it indispensable in today's dynamic business landscape.

**OVERVIEW OF THE PROJECT:**

* In this phase-2 of the project,we have innovated the design of the project (PRODUCT SALES ANALYSIS) using MACHINE LEARNING .
* Here we have taken “REC corp LTD. is small-scaled business venture established in India” as an example company and done the analysed the data of that company .
* Dataset link:[**https://www.kaggle.com/datasets/ksabishek/product-sales-data**](https://www.kaggle.com/datasets/ksabishek/product-sales-data)

INPUT[1]: *# import the important packages*

import pandas as pd *# library used for data manipulation and analysis*

import numpy as np *# library used for working with arrays* import matplotlib.pyplot as plt *# library for plots and visualizations*

import seaborn as sns *# library for visualizations*

%matplotlib inline

*# To ignore warnings*

import warnings warnings.filterwarnings("ignore")

INPUT[2]:#*if you open in Kaggle editor*

data = pd.read\_csv('/content/statsfinal.csv')

*#if you open in juypter notebook*

*# data = pd.read\_csv('statsfinal.csv')*

INPUT[3]:*# Checking the first 5 and last 5 rows of the dataset*

data.head(-1)

OUTPUT 3

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

P2 \

0 0 13-06-2010 5422 3725 576 907 17187.74

23616.50

1 1 14-06-2010 7047 779 3578 1574 22338.99

4938.86

2 2 15-06-2010 1572 2082 595 1145 4983.24

13199.88

3 3 16-06-2010 5657 2399 3140 1672 17932.69

15209.66

4 4 17-06-2010 3668 3207 2184 708 11627.56

20332.38

... ... ... ... ... ... ... ... .

..

4594 4594 29-01-2023 1227 3044 5510 1896 3889.59

19298.96

4595 4595 30-01-2023 2476 3419 525 1359 7848.92

21676.46

4596 4596 31-01-2023 7446 841 4825 1311 23603.82

5331.94

4597 4597 01-02-2023 6289 3143 3588 474 19936.13

19926.62

4598 4598 02-02-2023 3122 1188 5899 517 9896.74

7531.92

|  |  |  |
| --- | --- | --- |
|  | S-P3 | S-P4 |
| 0 | 3121.92 | 6466.91 |
| 1 | 19392.76 | 11222.62 |
| 2 | 3224.90 | 8163.85 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 3 | 17018.80 | 11921.36 |
| 4 | 11837.28 | 5048.04 |
| ... | ... | ... |
| 4594 | 29864.20 | 13518.48 |
| 4595 | 2845.50 | 9689.67 |
| 4596 | 26151.50 | 9347.43 |
| 4597 | 19446.96 | 3379.62 |
|  | 4598 | 31972.58 | 3686.21 |

[4599 rows x 10 columns]

INPUT[4]:*# drop the first column*

data = data.drop(columns=['Unnamed: 0']) data.info()

INPUT[5]:<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

[6]data.isnull().s

OUTPUT[6]:

um() 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

INPUT[7] *# Extract year from the 'Day' 'Month' 'year' from the 'Date' column using a lambda function*

*# We need to get the year from the data to analyse sales year to year*

data['Day'] = data['Date'].apply(lambda x: x.split('-')[0])

data['Month'] = data['Date'].apply(lambda x: x.split('-')[1])

OUTPUT[7]:

data['Year'] = data['Date'].apply(lambda x: x.split('-')[2]) data

Date Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 S-P3

\

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

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

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

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

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

... ... ... ... ... ... ... ... ...

4595 30-01-2023 2476 3419 525 1359 7848.92 21676.46 2845.50

4596 31-01-2023 7446 841 4825 1311 23603.82 5331.94 26151.50

4597 01-02-2023 6289 3143 3588 474 19936.13 19926.62 19446.96

4598 02-02-2023 3122 1188 5899 517 9896.74 7531.92 31972.58

4599 03-02-2023 1234 3854 2321 406 3911.78 24434.36 12579.82

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | S-P4 | Day | Month | Year |
| 0 | 6466.91 | 13 | 06 | 2010 |
| 1 | 11222.62 | 14 | 06 | 2010 |
| 2 | 8163.85 | 15 | 06 | 2010 |
| 3 | 11921.36 | 16 | 06 | 2010 |
| 4 | 5048.04 | 17 | 06 | 2010 |
| ... | ... | .. | ... | ... |
| 4595 | 9689.67 | 30 | 01 | 2023 |
| 4596 | 9347.43 | 31 | 01 | 2023 |
| 4597 | 3379.62 | 01 | 02 | 2023 |
| 4598 | 3686.21 | 02 | 02 | 2023 |
| 4599 | 2894.78 | 03 | 02 | 2023 |

[4600 rows x 12 columns]

INPUT[8]

data\_reduced = data.query("Year != '2010' and Year != '2023'")

INPUT[9]

*#Create a function that allows us to plot a bar chart for the 4 products*

def plot\_bar\_chart(df, columns, stri, str1, val):

*# Aggregate sales for each product by year, by sum or mean*

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()

*# Melt the data to make it easier to plot*

sales\_by\_year\_melted = pd.melt(sales\_by\_year, id\_vars='Year', value\_vars=columns, var\_name='Product', value\_name='Sales')

*# Create a bar chart*

plt.figure(figsize=(20,4)) sns.barplot(data=sales\_by\_year\_melted, x='Year', y='Sales',

hue='Product') *#,palette="cividis")* plt.xlabel('Year') plt.ylabel(stri) plt.title(f'{stri} by {str1}') plt.xticks(rotation=45) plt.show()

INPUT[10]:

*#use the plot\_bar\_chart function, enter the Unit Sales Columns and the Unit Sales string*

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')

