Project Title: Product Sales Analysis

PRADEEPKUMAR R ( r.pradeep40534@gmail.com)

MANIKANDAN K (manikandan958512@gmail.com)

SHANMUGAM S ( shanmugam007143@gmail.com )

PRAVEEN KUMAR D ( praveenares01@gmail.com )

GOKUL P ( masgokul05@gmail.com )

Product sales analysis is the process of evaluating and interpreting data related to the sales of products to gain insights into performance, trends, and opportunities for improvement. It involves examining various aspects of sales data, such as sales revenue, quantity sold, product categories, and customer behavior. The main objectives of product sales analysis are to:

**Understand Performance:**

Assess how well products are selling, which products are top performers, and which are underperforming.

**Identify Trends:**

Analyze historical sales data to uncover trends over time, such as seasonality or changing customer preferences.

**Forecasting:**

Use past sales data to make predictions about future sales, enabling inventory management and resource allocation.

**Customer Insights:**

Understand customer behavior, preferences, and buying patterns to tailor marketing and sales strategies.

**Optimize Pricing:**

Determine if pricing adjustments can maximize profitability or market share. Inventory Management: Manage stock levels efficiently by identifying slow-moving or overstocked products. Marketing and Promotion Evaluation: Assess the impact of marketing campaigns and promotions on sales.

**Importing libraries :**

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

**Importing data :**

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

df.head(5)

output:



## **Workflow**

* Understanding the data
* Data cleaning

Understanding the data

df.shape

o/p:

(4600, 10)

*# fetching column names*

df.columns

o/p :

Index(['Unnamed: 0', 'Date', 'Q-P1', 'Q-P2', 'Q-P3', 'Q-P4', 'S-P1', 'S-P2',

'S-P3', 'S-P4'],

dtype='object')

basic info:

df.info()

o/p:

<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

*Checking null values*

df.isnull().sum()

O/p:

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

#### No Null values

Finding duplicates:

df.duplicated().sum()

O/p

0

DATA CLEANING

*Changing dtype*

from datetime import datetime as dt

df[df["Date"]=="31-9-2010"]

O/P:



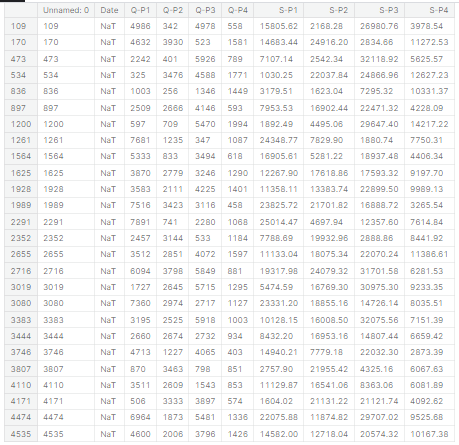
Handling Missing Data:

Identify and address missing values. Depending on the nature and extent of missing data,

df['Date'] = pd.to\_datetime(df['Date'], errors='coerce')

df[df['Date'].isnull()]

O/P:



Removing null values:

Remove rows with missing values if they are a small portion of the dataset.

Impute missing values using methods like mean, median, or machine learning

algorithms.

df['Date'].isnull().sum()

O/P:

0

Identifing datatypes :

df.dtypes

O/P:

Unnamed: 0 int64

Date datetime64[ns]

Q-P1 int64

Q-P2 int64

Q-P3 int64

Q-P4 int64

S-P1 float64

S-P2 float64

S-P3 float64

S-P4 float64

dtype: object

*fetching month,day of week, weekday*

df["month"]=df["Date"].dt.month\_name()

df["day"]=df["Date"].dt.day\_name()

df["dayoftheweek"]=df["Date"].dt.weekday

df["year"]=df["Date"].dt.year

df.sample()

O/P:



*Droping column unnamed as it is not usefull for us*

df.drop(columns=["Unnamed: 0"],inplace=True)

df.sample()

O/P:



*# Total unit sales Product 1, Product 2, Product 3, Product 4*

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])

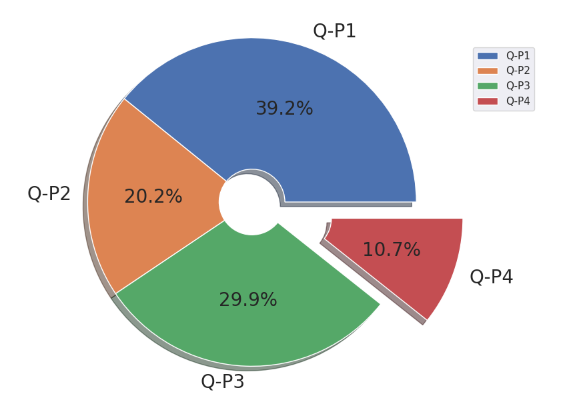
plt.legend(loc='center right', bbox\_to\_anchor=(1.2, 0.8));

Q-P1 18960506

Q-P2 9799295

Q-P3 14470404

Q-P4 5168100



*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

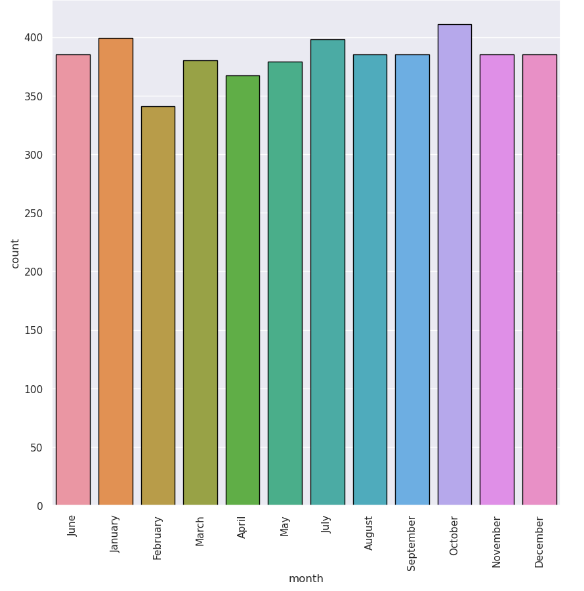
March 380

May 379

April 367

February 341

Name: month, dtype: int64



*which is the most occuring Day*

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

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

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

plt.xticks(rotation=90);

Friday 680

Wednesday 655

Sunday 654

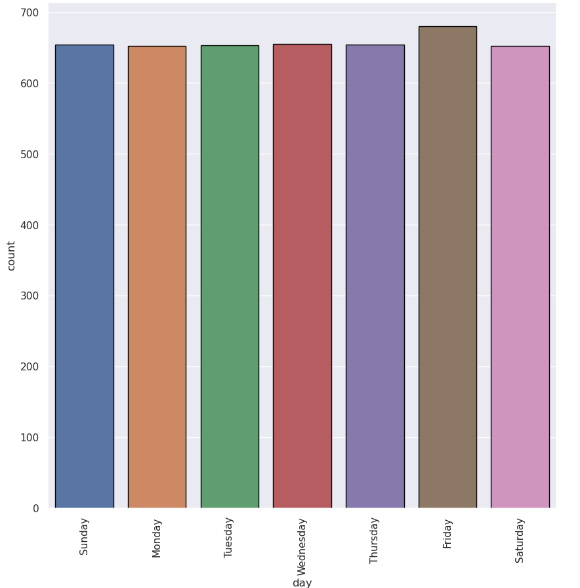
Thursday 654

Tuesday 653

Monday 652

Saturday 652

Name: day, dtype: int64



*which is the most occuring year*

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

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

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

plt.xticks(rotation=90);

2016 387

2011 362

2013 362

2014 362

2015 362

2017 362

2018 362

2019 362

2021 362

2022 362

2012 361

2020 361

2010 199

2023 34

Name: year, dtype: int64

