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:

	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

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:

	Unnamed: 0	Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4
109	109	31-9-2010	4986	342	4978	558	15805.62	2168.28	26980.76	3978.54

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:

	Unnamed: 0	Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4
109	109	NaT	4986	342	4978	558	15805.62	2168.28	26980.76	3978.54
170	170	NaT	4632	3930	523	1581	14683.44	24916.20	2834.66	11272.53
473	473	NaT	2242	401	5926	789	7107.14	2542.34	32118.92	5625.57
534	534	NaT	325	3476	4588	1771	1030.25	22037.84	24866.96	12627.23
836	836	NaT	1003	256	1346	1449	3179.51	1623.04	7295.32	10331.37
897	897	NaT	2509	2666	4146	593	7953.53	16902.44	22471.32	4228.09
1200	1200	NaT	597	709	5470	1994	1892.49	4495.06	29647.40	14217.22
1261	1261	NaT	7681	1235	347	1087	24348.77	7829.90	1880.74	7750.31
1564	1564	NaT	5333	833	3494	618	16905.61	5281.22	18937.48	4406.34
1625	1625	NaT	3870	2779	3246	1290	12267.90	17618.86	17593.32	9197.70
1928	1928	NaT	3583	2111	4225	1401	11358.11	13383.74	22899.50	9989.13
1989	1989	NaT	7516	3423	3116	458	23825.72	21701.82	16888.72	3265.54
2291	2291	NaT	7891	741	2280	1068	25014.47	4697.94	12357.60	7614.84
2352	2352	NaT	2457	3144	533	1184	7788.69	19932.96	2888.86	8441.92
2655	2655	NaT	3512	2851	4072	1597	11133.04	18075.34	22070.24	11386.61
2716	2716	NaT	6094	3798	5849	881	19317.98	24079.32	31701.58	6281.53
3019	3019	NaT	1727	2645	5715	1295	5474.59	16769.30	30975.30	9233.35
3080	3080	NaT	7360	2974	2717	1127	23331.20	18855.16	14726.14	8035.51
3383	3383	NaT	3195	2525	5918	1003	10128.15	16008.50	32075.56	7151.39
3444	3444	NaT	2660	2674	2732	934	8432.20	16953.16	14807.44	6659.42
3746	3746	NaT	4713	1227	4065	403	14940.21	7779.18	22032.30	2873.39
3807	3807	NaT	870	3463	798	851	2757.90	21955.42	4325.16	6067.63
4110	4110	NaT	3511	2609	1543	853	11129.87	16541.06	8363.06	6081.89
4171	4171	NaT	506	3333	3897	574	1604.02	21131.22	21121.74	4092.62
4474	4474	NaT	6964	1873	5481	1336	22075.88	11874.82	29707.02	9525.68
4535	4535	NaT	4600	2006	3796	1426	14582.00	12718.04	20574.32	10167.38

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:

	Unnamed: 0	Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4	month	day	dayoftheweek	year
3465	3465	2019-12-21	5308	3408	2721	1262	16826.36	21606.72	14747.82	8998.06	December	Saturday	5	2019

Droping column unnamed as it is not usefull for us

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

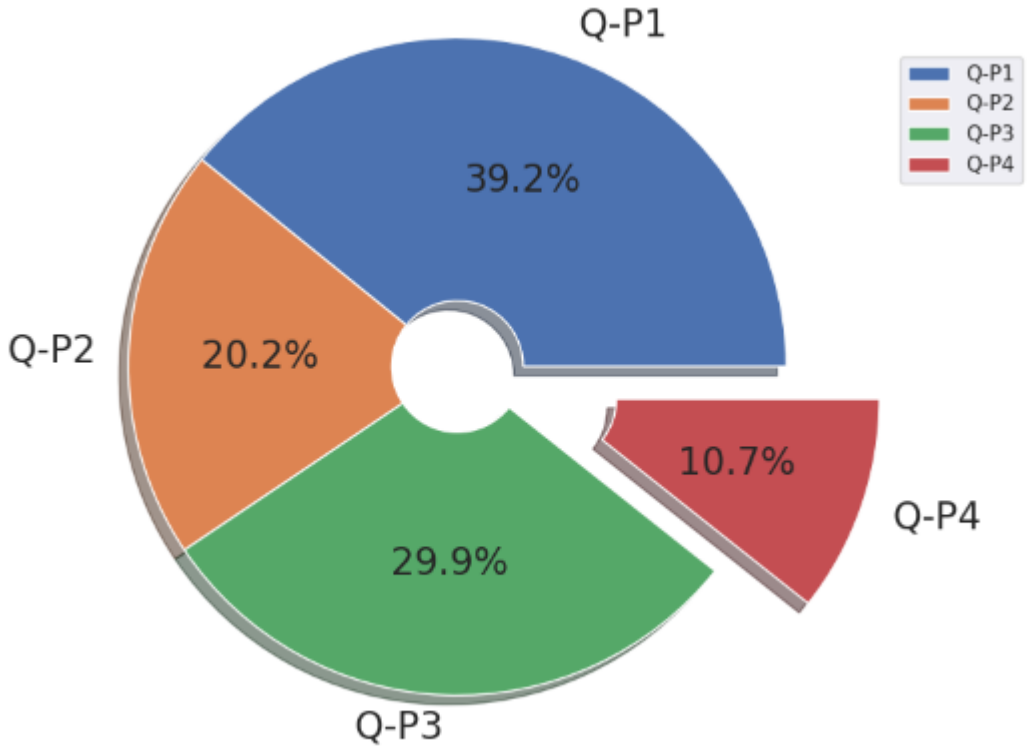
O/P:

	Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4	month	day	dayoftheweek	year
2978	2018-08-20	3047	3768	5824	1721	9658.99	23889.12	31566.08	12270.73	August	Monday	0	2018

 # 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
```

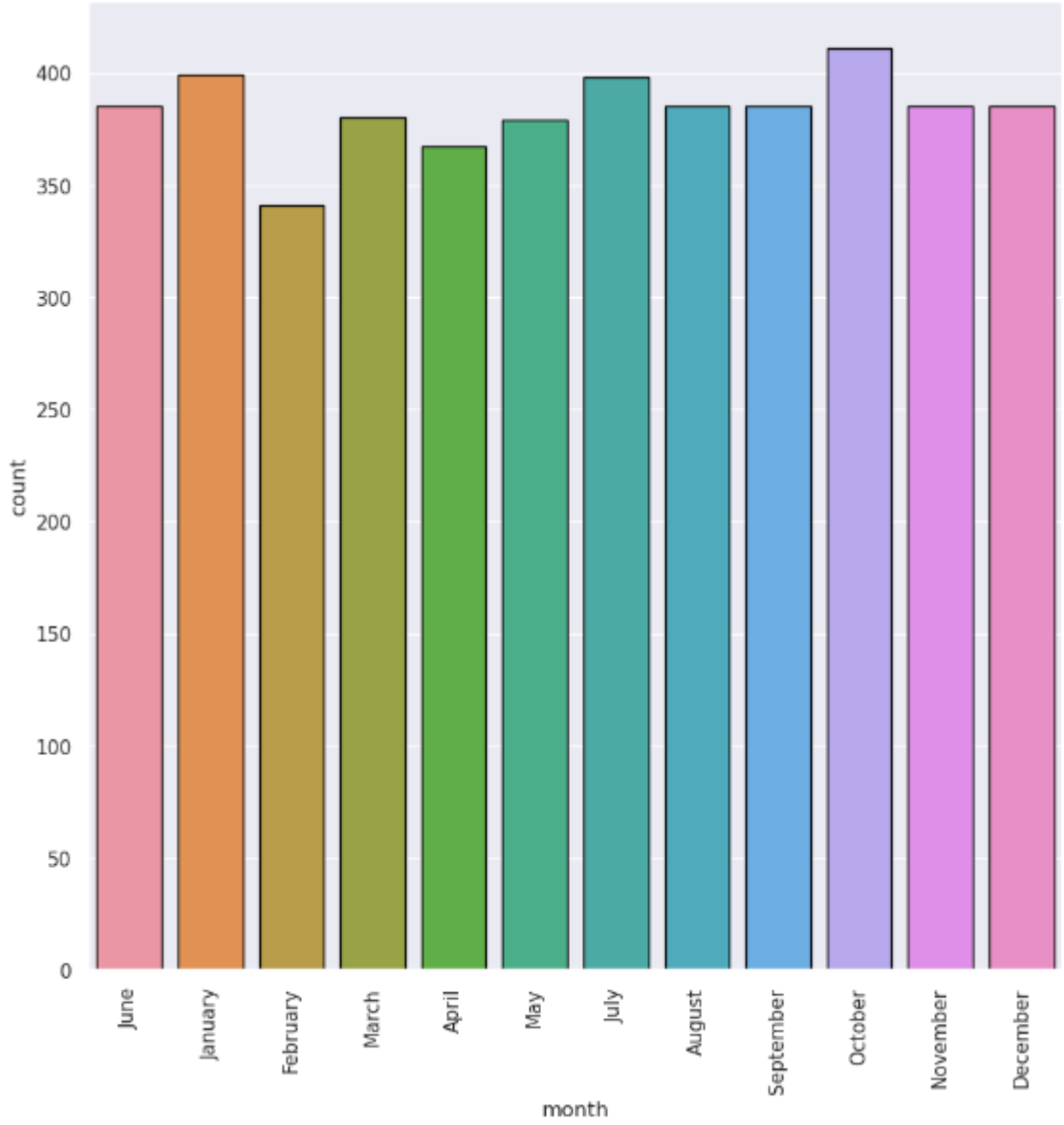


which is the most occuring month

```
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
```

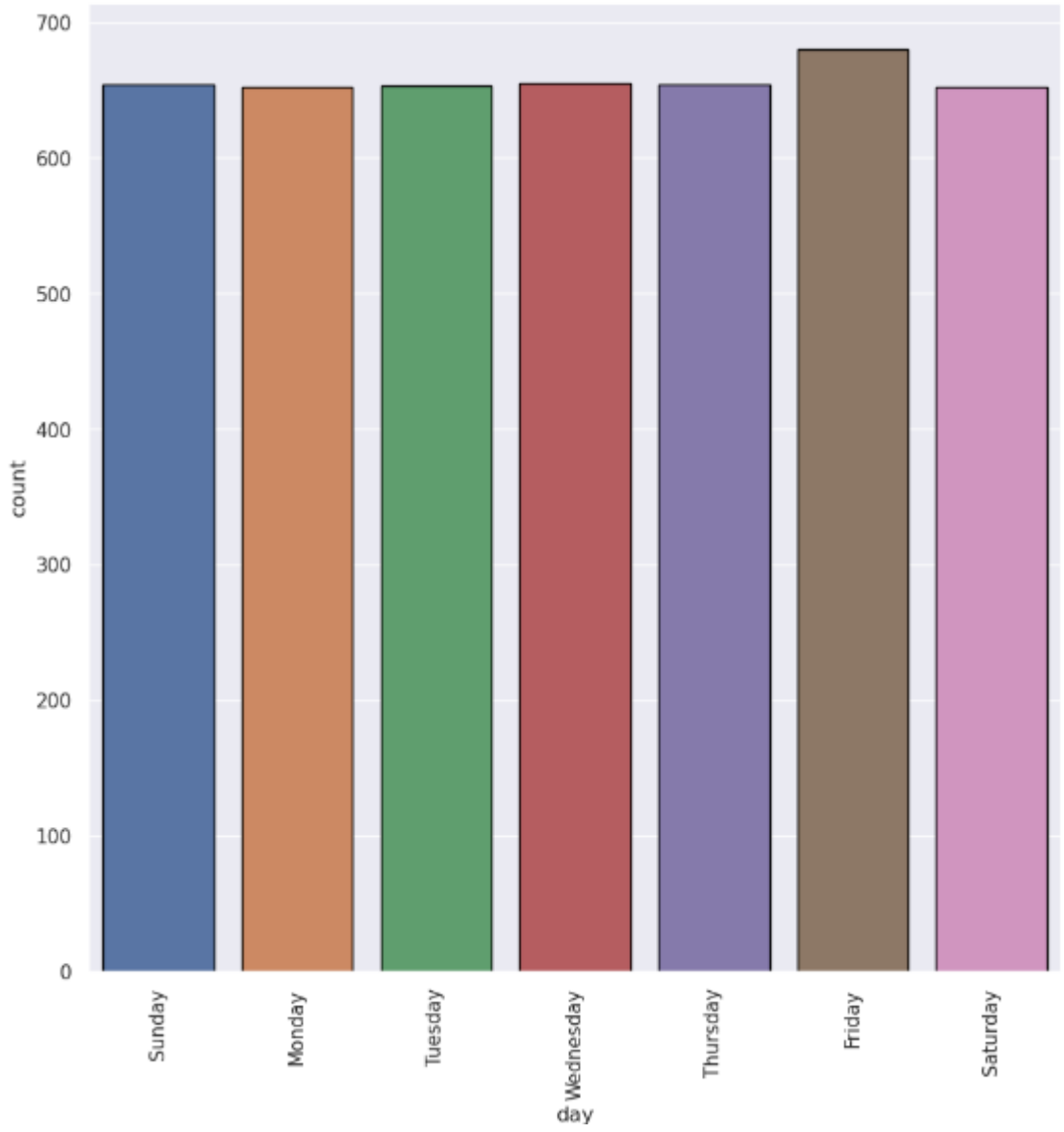
```
September    385
November     385
December     385
March        380
May          379
April        367
February     341
Name: month, dtype: int64
```



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
```