phase4

October 25, 2023

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
# Load the dataset
df = pd.read_csv('C:/Users/Senthil/Documents/product/statsfinal.csv')
```

[3]:
```
print(df.head())
```

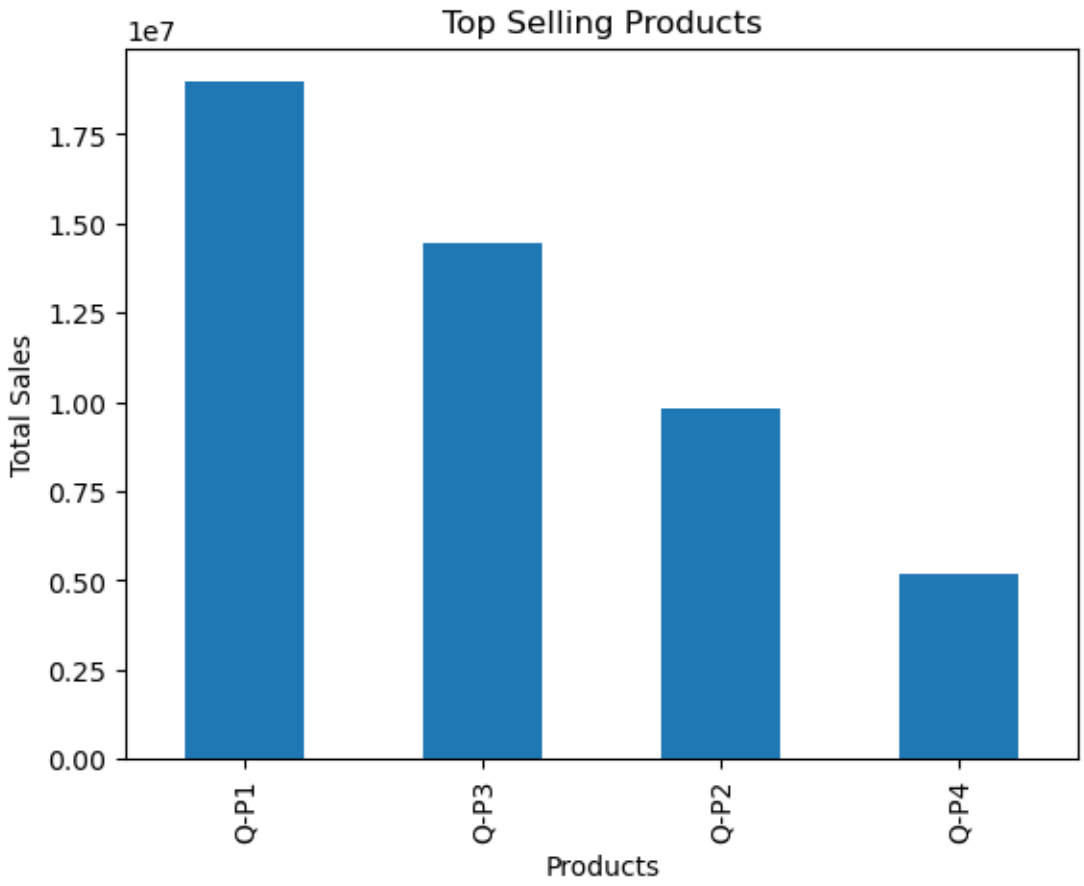
```
   Unnamed: 0        Date  Q-P1  Q-P2  Q-P3  Q-P4      S-P1      S-P2  \
0           0  13-06-2010  5422  3725   576   907  17187.74  23616.50
1           1  14-06-2010  7047   779  3578  1574  22338.99   4938.86
2           2  15-06-2010  1572  2082   595  1145   4983.24  13199.88
3           3  16-06-2010  5657  2399  3140  1672  17932.69  15209.66
4           4  17-06-2010  3668  3207  2184   708  11627.56  20332.38

       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
```

[4]:
```
# Check if 'Date' column can be converted to datetime
try:
    df['Date'] = pd.to_datetime(df['Date'], format='%d-%m-%Y')
except ValueError as e:
    print(f"Error: {e}")

# Identify and handle rows with invalid dates
invalid_dates = df[df['Date'].apply(lambda x: not isinstance(x, pd.Timestamp))]

# Print out the rows with problematic dates
print(invalid_dates)
```

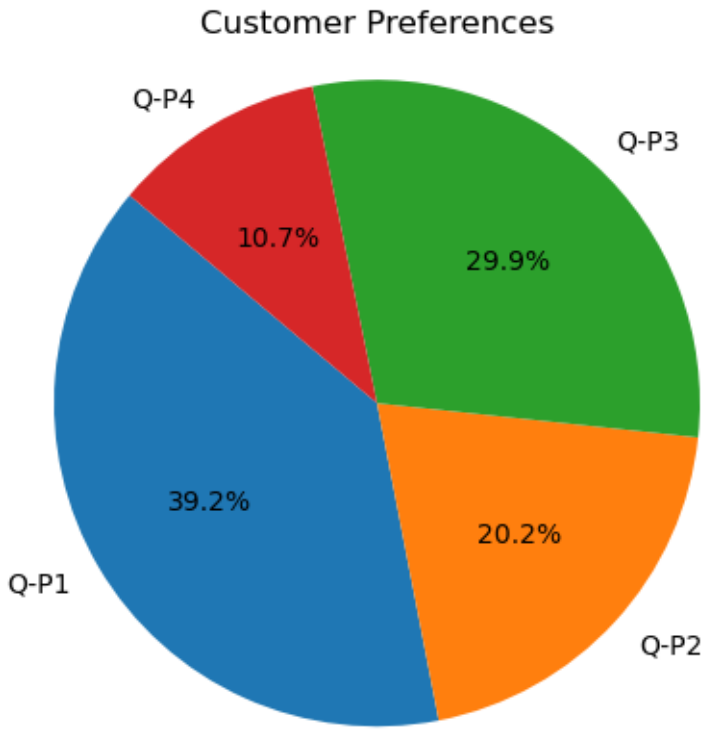
```
Error: day is out of range for month
      Unnamed: 0        Date  Q-P1  Q-P2  Q-P3  Q-P4      S-P1      S-P2  \
0              0  13-06-2010  5422  3725   576   907  17187.74  23616.50
```

```
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
…            …          …    …    …    …    …       …        …
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
4599        4599  03-02-2023  1234  3854  2321   406   3911.78  24434.36

          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
…          …         …
4595   2845.50   9689.67
4596  26151.50   9347.43
4597  19446.96   3379.62
4598  31972.58   3686.21
4599  12579.82   2894.78

[4600 rows x 10 columns]
```

[5]:
```
# 3.1. Top-Selling Products
top_products = df[['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']].sum().
  ↪sort_values(ascending=False)
top_products.plot(kind='bar', title='Top Selling Products')
plt.xlabel('Products')
plt.ylabel('Total Sales')
plt.show()
```



[6]:
```
product_sum = df[['Q-P1', 'Q-P2', 'Q-P3', 'Q-P4']].sum()
plt.pie(product_sum, labels=product_sum.index, autopct='%1.1f%%',␣
 ↪startangle=140)
plt.axis('equal')
plt.title('Customer Preferences')
plt.show()
```


Customer Preferences
Q-P4
Q-P3
10.7%
29.9%
39.2%
20.2%
Q-P1
Q-P2

[ ]: