Untitled2

May 13, 2025

[1]:
```
import pandas as pd
```

[3]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[9]:
```
df = pd.read_csv(r'C:\Users\TOP10\Desktop\New folder\SuperMarket Analysis.csv')
```

[11]:
```
df.head()
```

[11]:

	Invoice ID	Branch	City	Customer type	Gender	\
0	750-67-8428	Alex	Yangon	Member	Female	
1	226-31-3081	Giza	Naypyitaw	Normal	Female	
2	631-41-3108	Alex	Yangon	Normal	Female	
3	123-19-1176	Alex	Yangon	Member	Female	
4	373-73-7910	Alex	Yangon	Member	Female	

	Product line	Unit price	Quantity	Tax 5%	Sales	Date	\
0	Health and beauty	74.69	7	26.1415	548.9715	1/5/2019	
1	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2019	
2	Home and lifestyle	46.33	7	16.2155	340.5255	3/3/2019	
3	Health and beauty	58.22	8	23.2880	489.0480	1/27/2019	
4	Sports and travel	86.31	7	30.2085	634.3785	2/8/2019	

	Time	Payment	cogs	gross margin percentage	gross income	\
0	1:08:00 PM	Ewallet	522.83	4.761905	26.1415	
1	10:29:00 AM	Cash	76.40	4.761905	3.8200	
2	1:23:00 PM	Credit card	324.31	4.761905	16.2155	
3	8:33:00 PM	Ewallet	465.76	4.761905	23.2880	
4	10:37:00 AM	Ewallet	604.17	4.761905	30.2085	

	Rating
0	9.1
1	9.6
2	7.4
3	8.4
4	5.3

[13]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Invoice ID               1000 non-null   object
 1   Branch                   1000 non-null   object
 2   City                     1000 non-null   object
 3   Customer type            1000 non-null   object
 4   Gender                   1000 non-null   object
 5   Product line             1000 non-null   object
 6   Unit price               1000 non-null   float64
 7   Quantity                 1000 non-null   int64
 8   Tax 5%                   1000 non-null   float64
 9   Sales                    1000 non-null   float64
 10  Date                     1000 non-null   object
 11  Time                     1000 non-null   object
 12  Payment                  1000 non-null   object
 13  cogs                     1000 non-null   float64
 14  gross margin percentage  1000 non-null   float64
 15  gross income             1000 non-null   float64
 16  Rating                   1000 non-null   float64
dtypes: float64(7), int64(1), object(9)
memory usage: 132.9+ KB
```

[19]:
```
df.describe()
```

[19]:

	Unit price	Quantity	Tax 5%	Sales	cogs	\
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000	
mean	55.672130	5.510000	15.379369	322.966749	307.58738	
std	26.494628	2.923431	11.708825	245.885335	234.17651	
min	10.080000	1.000000	0.508500	10.678500	10.17000	
25%	32.875000	3.000000	5.924875	124.422375	118.49750	
50%	55.230000	5.000000	12.088000	253.848000	241.76000	
75%	77.935000	8.000000	22.445250	471.350250	448.90500	
max	99.960000	10.000000	49.650000	1042.650000	993.00000	

	gross margin percentage	gross income	Rating
count	1.000000e+03	1000.000000	1000.00000
mean	4.761905e+00	15.379369	6.97270
std	6.131498e-14	11.708825	1.71858
min	4.761905e+00	0.508500	4.00000
25%	4.761905e+00	5.924875	5.50000
50%	4.761905e+00	12.088000	7.00000
75%	4.761905e+00	22.445250	8.50000
max	4.761905e+00	49.650000	10.00000

[21]:
```
df.isnull().sum()
```

[21]:
```
Invoice ID                 0
Branch                     0
City                       0
Customer type              0
Gender                     0
Product line               0
Unit price                 0
Quantity                   0
Tax 5%                     0
Sales                      0
Date                       0
Time                       0
Payment                    0
cogs                       0
gross margin percentage    0
gross income               0
Rating                     0
dtype: int64
```

[23]:
```
df.duplicated().sum()
```

[23]:
```
0
```

[29]:
```
df.describe()
df['Payment'].value_counts()
df['Product line'].unique()
```

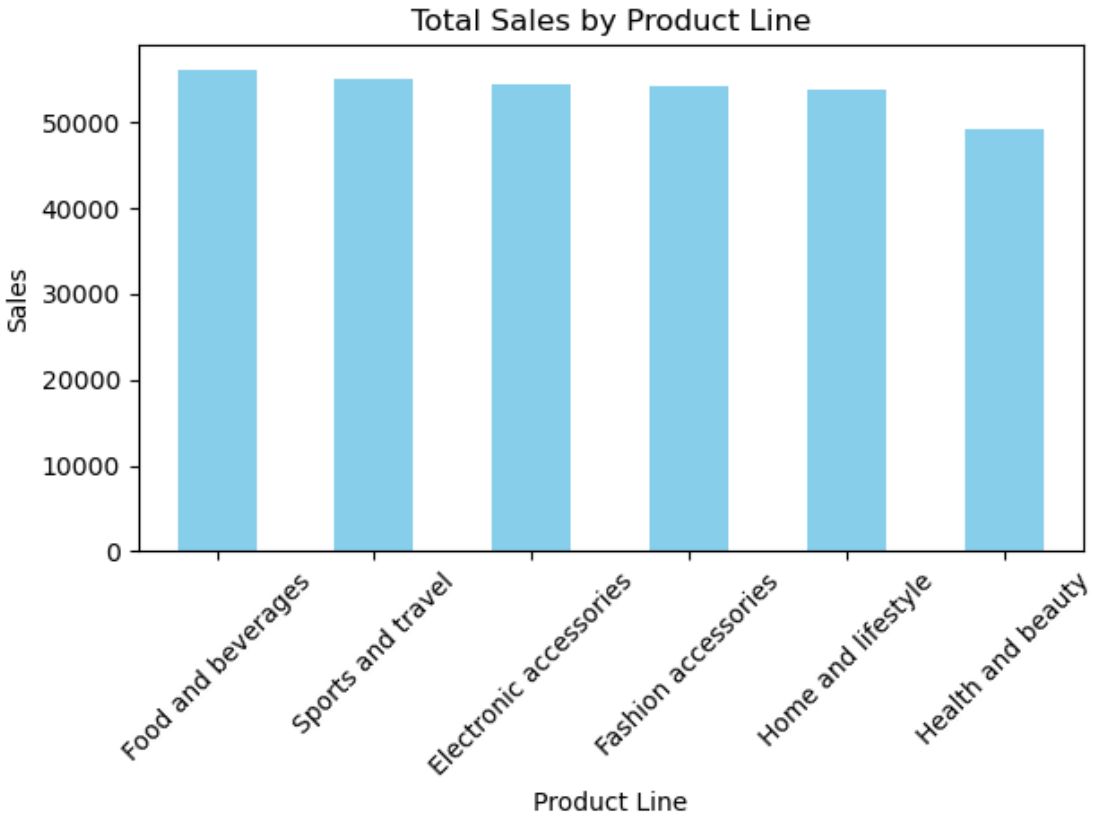
[29]:
```
array(['Health and beauty', 'Electronic accessories',
       'Home and lifestyle', 'Sports and travel', 'Food and beverages',
       'Fashion accessories'], dtype=object)
```

[31]:
```
sales_by_product = df.groupby('Product line')['Sales'].sum().
 ↪sort_values(ascending=False)
print(sales_by_product)
```

```
Product line
Food and beverages        56144.8440
Sports and travel         55122.8265
Electronic accessories    54337.5315
Fashion accessories       54305.8950
Home and lifestyle        53861.9130
Health and beauty         49193.7390
Name: Sales, dtype: float64
```

[33]:
```
sales_by_product.plot(kind='bar', color='skyblue')
plt.title('Total Sales by Product Line')
```

```
plt.ylabel('Sales')
plt.xlabel('Product Line')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



[35]:
```
ratings = df.groupby('Product line')['Rating'].mean().
  ↪sort_values(ascending=False)
print(ratings)
```

```
Product line
Food and beverages         7.113218
Fashion accessories        7.029213
Health and beauty          7.003289
Electronic accessories     6.924706
Sports and travel          6.916265
Home and lifestyle         6.837500
Name: Rating, dtype: float64
```

[37]:
```
counts = df['Product line'].value_counts()
print(counts)
```

```
Product line
Fashion accessories       178
Food and beverages        174
Electronic accessories    170
Sports and travel         166
Home and lifestyle        160
Health and beauty         152
Name: count, dtype: int64
```

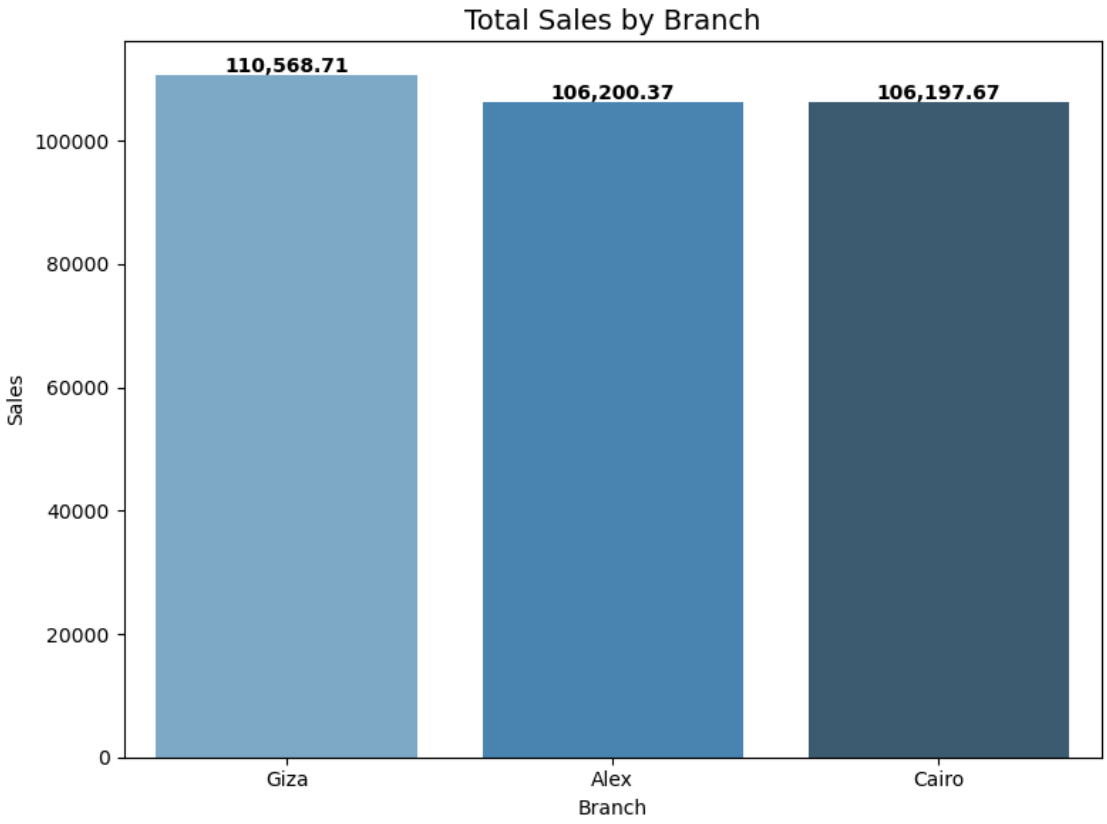
[41]:
```
total_sales = df['Sales'].sum()
print("total sales:", total_sales)
```

```
total sales: 322966.749
```

[ ]:
```
```

[45]:
```
import matplotlib.pyplot as plt
import seaborn as sns

branch_sales = df.groupby('Branch')['Sales'].sum().sort_values(ascending=False)

plt.figure(figsize=(8, 6))
ax = sns.barplot(x=branch_sales.index, y=branch_sales.values, palette='Blues_d')

for i, v in enumerate(branch_sales.values):
    ax.text(i, v + 500, f'{v:,.2f}', ha='center', fontweight='bold')

plt.title("Total Sales by Branch", fontsize=14)
plt.ylabel("Sales")
plt.xlabel("Branch")
plt.tight_layout()
plt.show()
```

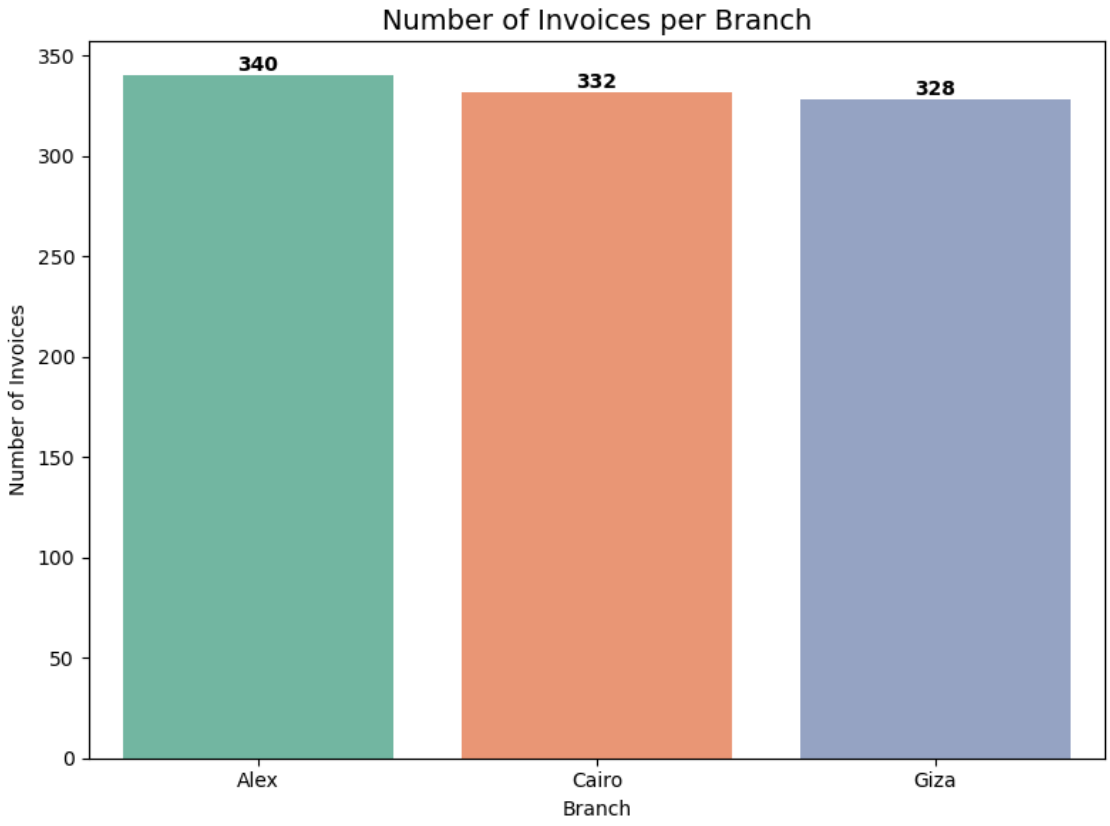
```
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\348827959.py:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  ax = sns.barplot(x=branch_sales.index, y=branch_sales.values,
palette='Blues_d')
```



[50]:
```
import matplotlib.pyplot as plt
import seaborn as sns

branch_counts = df['Branch'].value_counts()


plt.figure(figsize=(8, 6))
ax = sns.barplot(x=branch_counts.index, y=branch_counts.values, palette='Set2')


for i, v in enumerate(branch_counts.values):
    ax.text(i, v + 2, str(v), ha='center', fontweight='bold')


plt.title("Number of Invoices per Branch", fontsize=14)
plt.ylabel("Number of Invoices")
plt.xlabel("Branch")
plt.tight_layout()
plt.show()
```

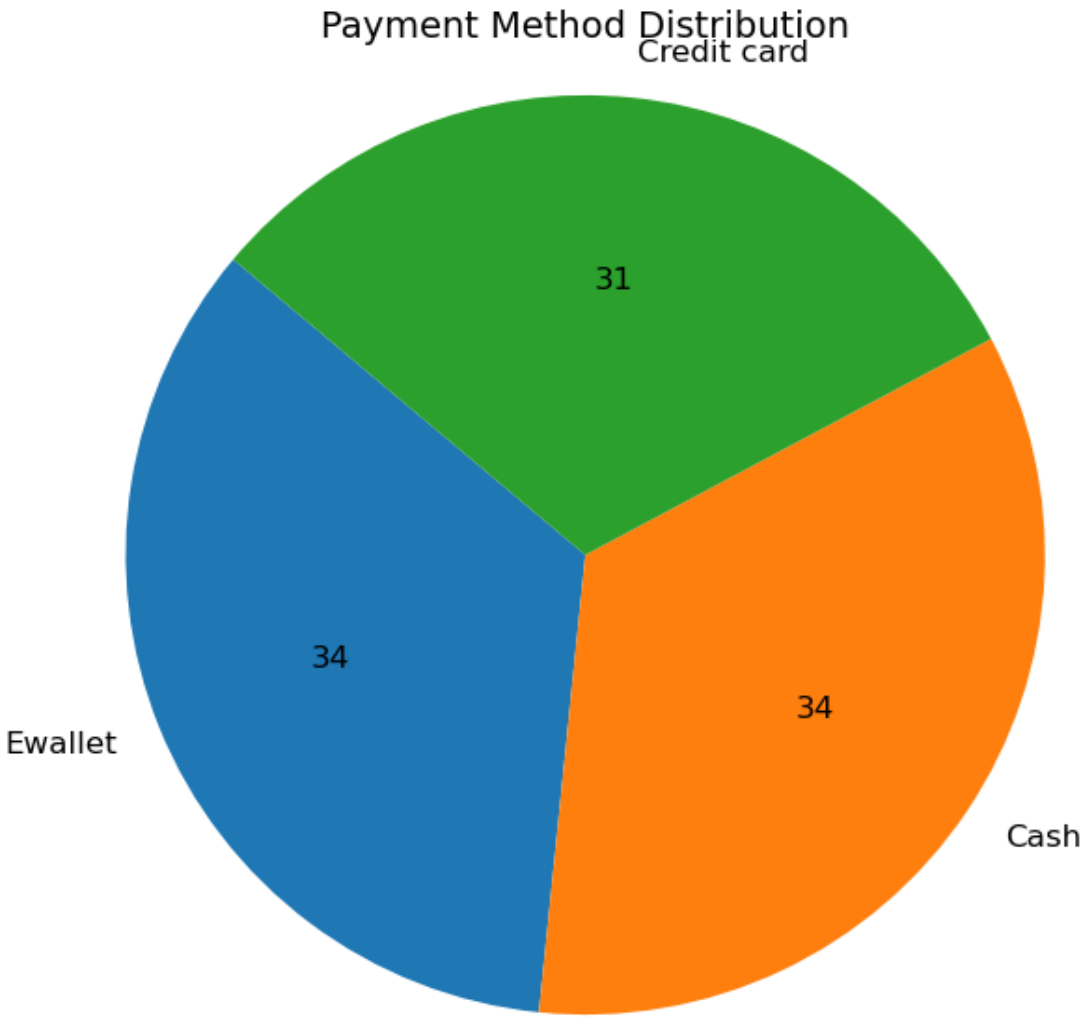
```
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\864553138.py:8: FutureWarning:
```

```
Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  ax = sns.barplot(x=branch_counts.index, y=branch_counts.values,
palette='Set2')
```



[54]:
```
import matplotlib.pyplot as plt


payment_counts = df['Payment'].value_counts()


plt.figure(figsize=(7, 7))
plt.pie(payment_counts, labels=payment_counts.index, autopct='%1.0f',␣
  ↪startangle=140, textprops={'fontsize': 12})


plt.title("Payment Method Distribution", fontsize=14)
plt.axis('equal')
```

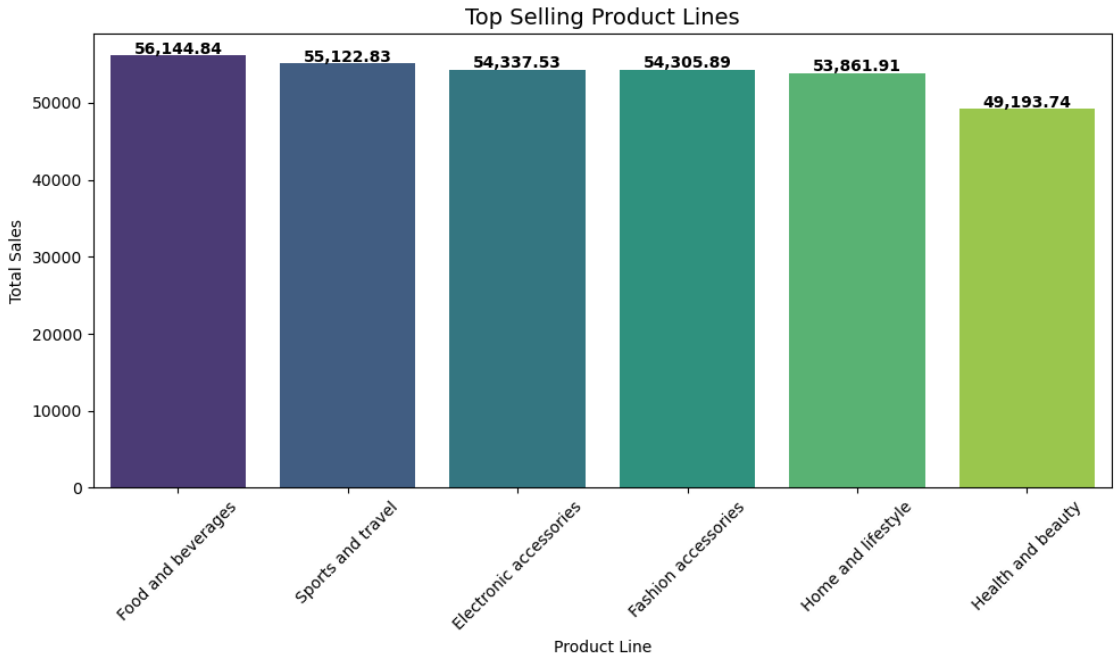
```
plt.show()
```



[58]:
```
import matplotlib.pyplot as plt
import seaborn as sns


product_sales = df.groupby('Product line')['Sales'].sum().
 ↪sort_values(ascending=False)


plt.figure(figsize=(10, 6))
ax = sns.barplot(x=product_sales.index, y=product_sales.values,␣
 ↪palette='viridis')
```

```
for i, v in enumerate(product_sales.values):
    ax.text(i, v + 200, f'{v:,.2f}', ha='center', fontweight='bold')


plt.title("Top Selling Product Lines", fontsize=14)
plt.ylabel("Total Sales")
plt.xlabel("Product Line")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

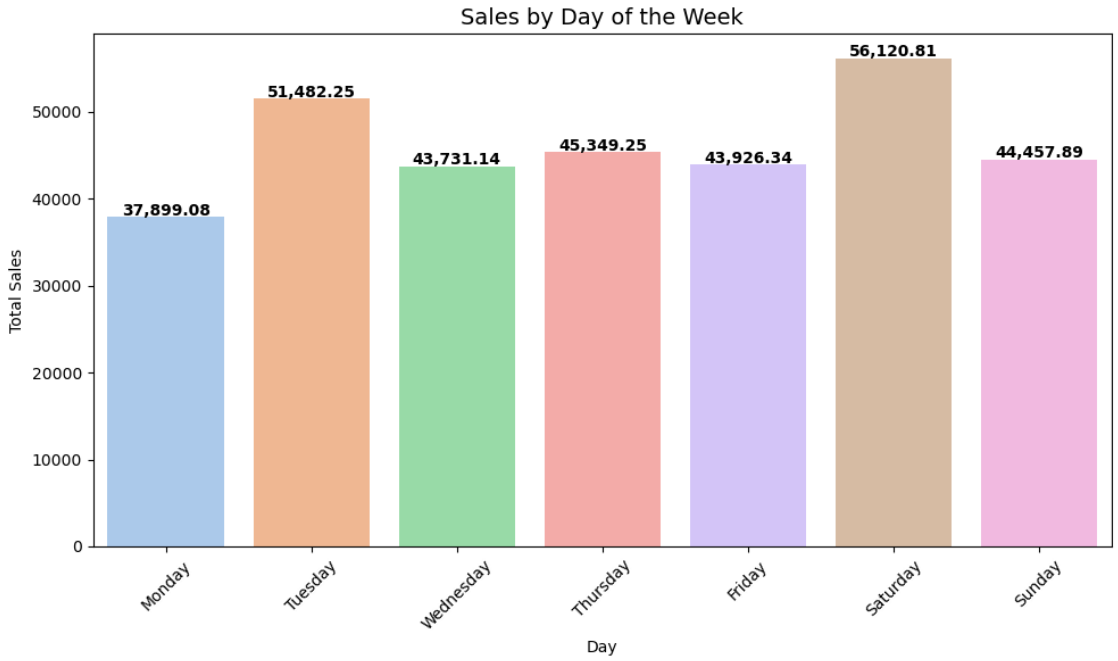
```
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\1926029675.py:9:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  ax = sns.barplot(x=product_sales.index, y=product_sales.values,
palette='viridis')
```



[60]:
```
df['Date'] = pd.to_datetime(df['Date'])
```

```
df['Day'] = df['Date'].dt.day_name()


days_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday',␣
 ↪'Saturday', 'Sunday']


sales_by_day = df.groupby('Day')['Sales'].sum().reindex(days_order)


plt.figure(figsize=(10, 6))
ax = sns.barplot(x=sales_by_day.index, y=sales_by_day.values, palette='pastel')


for i, v in enumerate(sales_by_day.values):
    ax.text(i, v + 200, f'{v:,.2f}', ha='center', fontweight='bold')


plt.title("Sales by Day of the Week", fontsize=14)
plt.ylabel("Total Sales")
plt.xlabel("Day")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

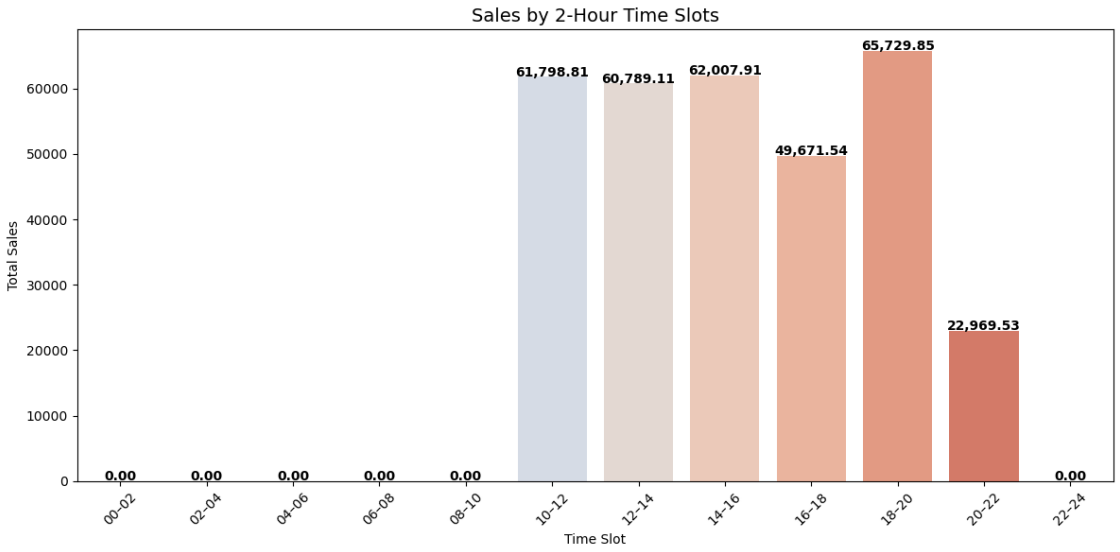
```
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\2505052615.py:14:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  ax = sns.barplot(x=sales_by_day.index, y=sales_by_day.values,
palette='pastel')
```



[64]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

#              datetime
df['Time'] = pd.to_datetime(df['Time'])

#
df['Hour'] = df['Time'].dt.hour

#
bins = list(range(0, 25, 2))  #   0    24
labels = [f'{str(i).zfill(2)}-{str(i+2).zfill(2)}' for i in bins[:-1]]

#
df['Time Slot'] = pd.cut(df['Hour'], bins=bins, labels=labels, right=False)

#
sales_by_slot = df.groupby('Time Slot')['Sales'].sum()

#
plt.figure(figsize=(12, 6))
ax = sns.barplot(x=sales_by_slot.index, y=sales_by_slot.values,␣
 ↪palette='coolwarm')
```

```
#
for i, v in enumerate(sales_by_slot.values):
    ax.text(i, v + 100, f'{v:,.2f}', ha='center', fontweight='bold',␣
 ↪fontsize=10)

#
plt.title("Sales by 2-Hour Time Slots", fontsize=14)
plt.xlabel("Time Slot")
plt.ylabel("Total Sales")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

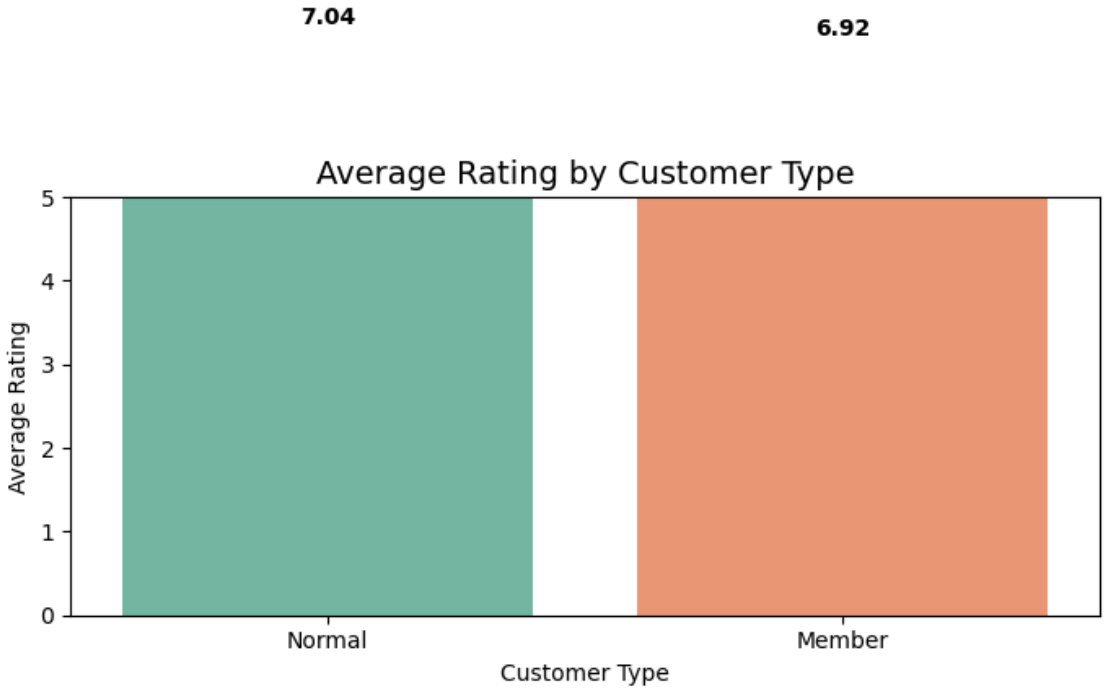
```
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\4252718816.py:6: UserWarning:
Could not infer format, so each element will be parsed individually, falling
back to `dateutil`. To ensure parsing is consistent and as-expected, please
specify a format.
  df['Time'] = pd.to_datetime(df['Time'])
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\4252718816.py:19:
FutureWarning: The default of observed=False is deprecated and will be changed
to True in a future version of pandas. Pass observed=False to retain current
behavior or observed=True to adopt the future default and silence this warning.
  sales_by_slot = df.groupby('Time Slot')['Sales'].sum()
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\4252718816.py:23:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  ax = sns.barplot(x=sales_by_slot.index, y=sales_by_slot.values,
palette='coolwarm')
```



[72]:
```
import matplotlib.pyplot as plt
import seaborn as sns

#
avg_rating_by_customer = df.groupby('Customer type')['Rating'].mean().
 ↪sort_values(ascending=False)

#
plt.figure(figsize=(7, 5))
ax = sns.barplot(x=avg_rating_by_customer.index, y=avg_rating_by_customer.
 ↪values, palette='Set2')

#
for i, v in enumerate(avg_rating_by_customer.values):
    ax.text(i, v + 0.02, f'{v:.2f}', ha='center', fontweight='bold')

#
plt.title("Average Rating by Customer Type", fontsize=14)
plt.ylabel("Average Rating")
plt.xlabel("Customer Type")
plt.ylim(0, 5)  #          5
plt.tight_layout()
plt.show()
```

```
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\3224636709.py:9:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
```

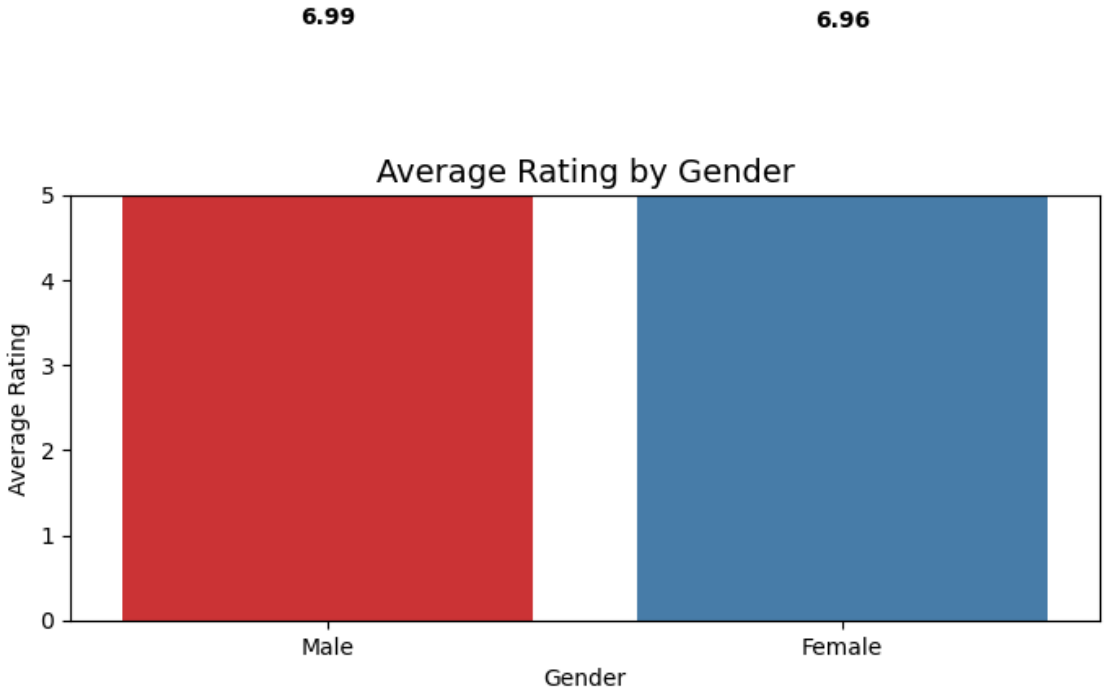
```
effect.

  ax = sns.barplot(x=avg_rating_by_customer.index,
y=avg_rating_by_customer.values, palette='Set2')
```



[74]:
```
import matplotlib.pyplot as plt
import seaborn as sns

#
avg_rating_by_gender = df.groupby('Gender')['Rating'].mean().
 ↪sort_values(ascending=False)

#
plt.figure(figsize=(7, 5))
ax = sns.barplot(x=avg_rating_by_gender.index, y=avg_rating_by_gender.values,␣
 ↪palette='Set1')

#
for i, v in enumerate(avg_rating_by_gender.values):
    ax.text(i, v + 0.02, f'{v:.2f}', ha='center', fontweight='bold')

#
plt.title("Average Rating by Gender", fontsize=14)
plt.ylabel("Average Rating")
```

```
plt.xlabel("Gender")
plt.ylim(0, 5)  #          5
plt.tight_layout()
plt.show()
```

```
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\885804705.py:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  ax = sns.barplot(x=avg_rating_by_gender.index, y=avg_rating_by_gender.values,
palette='Set1')
```



[76]:
```
import matplotlib.pyplot as plt
import seaborn as sns

#
avg_rating_by_gender = df.groupby('Gender')['Rating'].mean().
 ↪sort_values(ascending=False)

#
plt.figure(figsize=(7, 5))
```

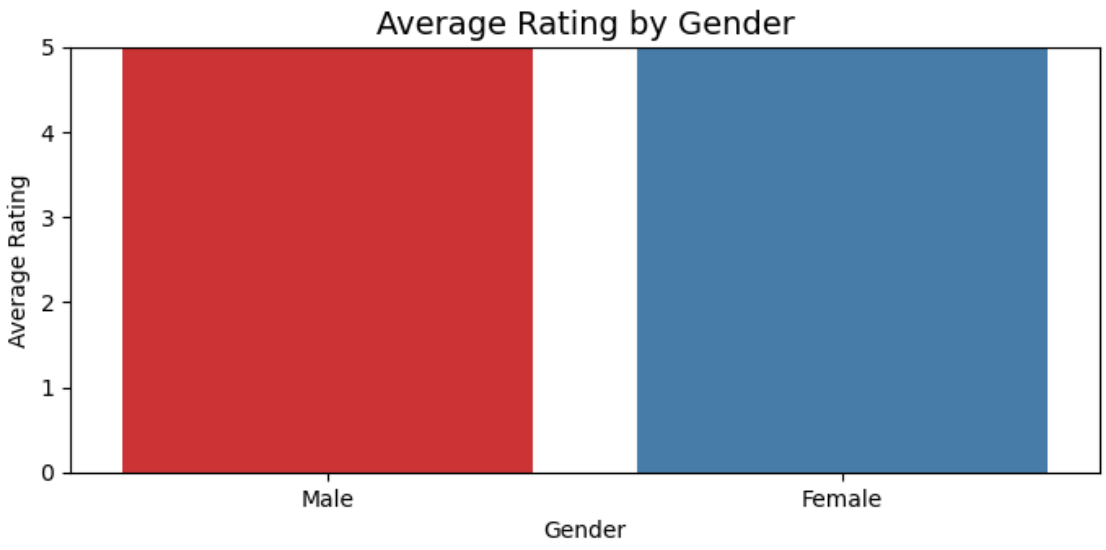
```
ax = sns.barplot(x=avg_rating_by_gender.index, y=avg_rating_by_gender.values,␣
 ↪palette='Set1')

#
for i, v in enumerate(avg_rating_by_gender.values):
    ax.text(i, v + 0.02, f'{v:.2f}', ha='center', fontweight='bold')

#
plt.title("Average Rating by Gender", fontsize=14)
plt.ylabel("Average Rating")
plt.xlabel("Gender")
plt.ylim(0, 5)  #           5
plt.tight_layout()

#
plt.savefig('avg_rating_by_gender.png', format='png')

#
report = f"""
                    :

                                                                    .


    :
1.              : {avg_rating_by_gender['Male']:.2f}
2.              : {avg_rating_by_gender['Female']:.2f}

                        'avg_rating_by_gender.png'.
"""

#
with open("final_report.txt", "w") as file:
    file.write(report)

#
plt.show()
```

```
C:\Users\TOP10\AppData\Local\Temp\ipykernel_12344\579307910.py:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  ax = sns.barplot(x=avg_rating_by_gender.index, y=avg_rating_by_gender.values,
palette='Set1')
```

```
--------------------------------------------------------------------------
```

```
UnicodeEncodeError                        Traceback (most recent call last)
Cell In[76], line 40
     38 #
     39 with open("final_report.txt", "w") as file:
---> 40     file.write(report)
     42 #
     43 plt.show()

File ~\anaconda3\Lib\encodings\cp1252.py:19, in IncrementalEncoder.encode(self,
 ↪input, final)
     18 def encode(self, input, final=False):
---> 19     return codecs.charmap_encode(input,self.errors,encoding_table)[0]

UnicodeEncodeError: 'charmap' codec can't encode characters in position 2-6:␣
 ↪character maps to <undefined>
```

6.99 **6.96**



[ ]: