```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
data = pd.read_csv('salesdata.csv',parse_dates=['Date'])
```

## Top 5 rows

```
data.head()
```

| | Invoice ID | Branch | City | Customer type | Gender | Product line | Unit price | Quantity | Tax 5% |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 750-67-8428 | A | Yangon | Member | Female | Health and beauty | 74.69 | 7 | 26.1415 |
| **1** | 226-31-3081 | C | Naypyitaw | Normal | Female | Electronic accessories | 15.28 | 5 | 3.8200 |
| **2** | 631-41-3108 | A | Yangon | Normal | Male | Home and lifestyle | 46.33 | 7 | 16.2155 |
| **3** | 123-19-1176 | A | Yangon | Member | Male | Health and beauty | 58.22 | 8 | 23.2880 |
| **4** | 373-73-7910 | A | Yangon | Normal | Male | Sports and travel | 86.31 | 7 | 30.2085 |

Next steps: Generate code with `data` | View recommended plots | New interactive sheet

## last 5 rows

```
data.tail()
```

| | Invoice ID | Branch | City | Customer type | Gender | Product line | Unit price | Quantity | Tax 5% |
|---|---|---|---|---|---|---|---|---|---|
| **995** | 233-67-5758 | C | Naypyitaw | Normal | Male | Health and beauty | 40.35 | 1 | 2.0175 |
| **996** | 303-96-2227 | B | Mandalay | Normal | Female | Home and lifestyle | 97.38 | 10 | 48.6900 |
| **997** | 727-02-1313 | A | Yangon | Member | Male | Food and beverages | 31.84 | 1 | 1.5920 |
| **998** | 347-56-2442 | A | Yangon | Normal | Male | Home and lifestyle | 65.82 | 1 | 3.2910 |
| **999** | 849-09-3807 | A | Yangon | Member | Female | Fashion accessories | 88.34 | 7 | 30.9190 |

random 5

```
data.sample(5)
```

| | Invoice ID | Branch | City | Customer type | Gender | Product line | Unit price | Quantity | Tax 5% | |
|---|---|---|---|---|---|---|---|---|---|---|
| **713** | 268-20-3585 | C | Naypyitaw | Normal | Female | Health and beauty | 13.85 | 9 | 6.2325 | 1 |
| **410** | 244-08-0162 | B | Mandalay | Normal | Female | Health and beauty | 34.21 | 10 | 17.1050 | 3 |
| **688** | 173-57-2300 | C | Naypyitaw | Member | Male | Sports and travel | 72.88 | 2 | 7.2880 | 1 |
| **696** | 182-52-7000 | A | Yangon | Member | Female | Sports and travel | 27.04 | 4 | 5.4080 | 1 |
| **93** | 152-08-9985 | B | Mandalay | Member | Male | Health and beauty | 64.36 | 9 | 28.9620 | 6 |

shape of data

```
print("total no columns",data.shape[0])
print("total no of rows",data.shape[1])
```

```
total no columns 1000
total no of rows 17
```

check for null values

```
data.isnull().sum()
```

```
                                 0
Invoice ID                       0
Branch                           0
City                             0
Customer type                    0
Gender                           0
Product line                     0
Unit price                       0
Quantity                         0
Tax 5%                           0
Total                            0
Date                             0
Time                             0
Payment                          0
cogs                             0
gross margin percentage          0
gross income                     0
Rating                           0

dtype: int64
```

to get over all statistics

```
data.describe()
```

| | Unit price | Quantity | Tax 5% | Total | Date | cogs |
|---|---|---|---|---|---|---|
| **count** | 1000.000000 | 1000.000000 | 1000.000000 | 1000.000000 | 1000 | 1000.00000 |
| **mean** | 55.672130 | 5.510000 | 15.379369 | 322.966749 | 2019-02-14 00:05:45.600000 | 307.58738 |
| **min** | 10.080000 | 1.000000 | 0.508500 | 10.678500 | 2019-01-01 00:00:00 | 10.17000 |
| **25%** | 32.875000 | 3.000000 | 5.924875 | 124.422375 | 2019-01-24 00:00:00 | 118.49750 |
| **50%** | 55.230000 | 5.000000 | 12.088000 | 253.848000 | 2019-02-13 00:00:00 | 241.76000 |
| **75%** | 77.935000 | 8.000000 | 22.445250 | 471.350250 | 2019-03-08 00:00:00 | 448.90500 |
| **max** | 99.960000 | 10.000000 | 49.650000 | 1042.650000 | 2019-03-30 00:00:00 | 993.00000 |
| **std** | 26.494628 | 2.923431 | 11.708825 | 245.885335 | NaN | 234.17651 |

## UNIVARIATE **ANALYSIS**

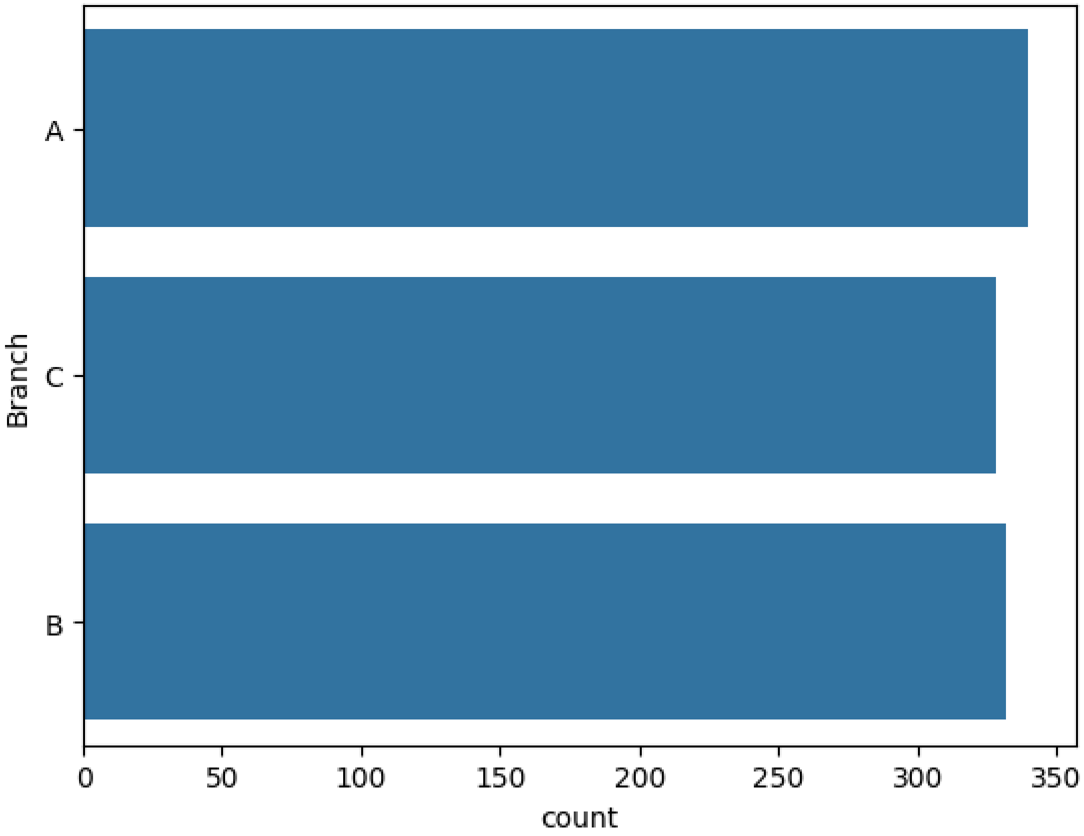
```
cat = []
num = []
for column in data.columns:
  if data[column].nunique() > 10:
    num.append(column)
  else:
    cat.append(column)
print("categorical;",cat)
print("numerical;",num)
```

```
categorical; ['Branch', 'City', 'Customer type', 'Gender', 'Product line', 'Quantity'
numerical; ['Invoice ID', 'Unit price', 'Tax 5%', 'Total', 'Date', 'Time', 'cogs', 'g
```

## aggregate sales among branches

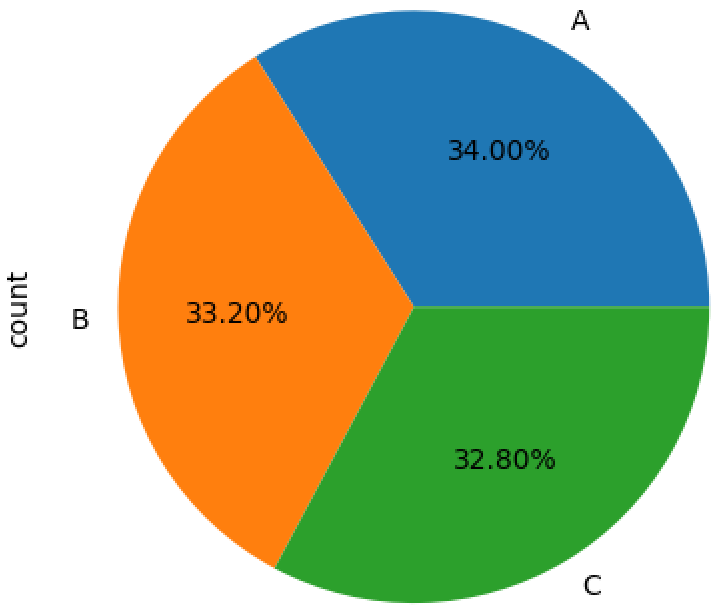
```
sns.countplot(data["Branch"])
```

```
<Axes: xlabel='count', ylabel='Branch'>
```



```
data["Branch"].value_counts().plot(kind = "pie",autopct="%1.2f%%")
```

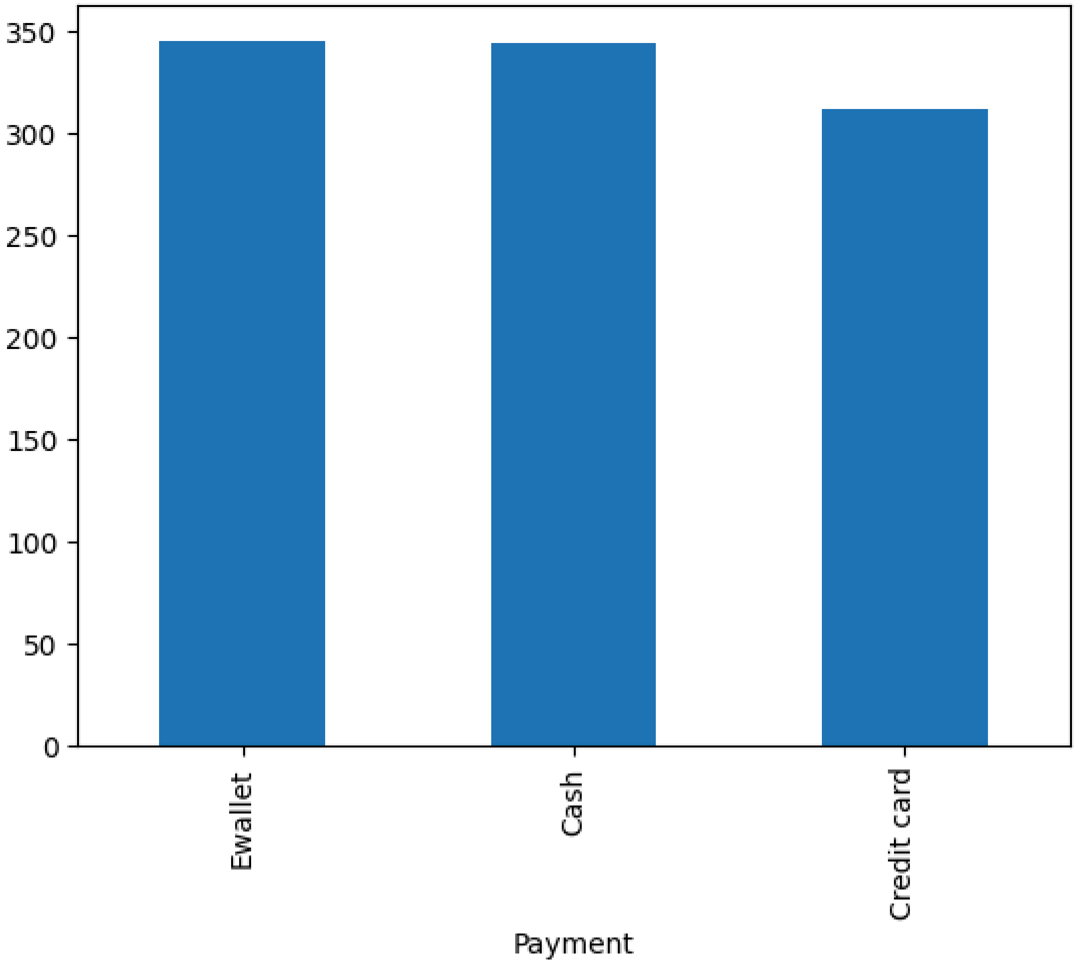
```
<Axes: ylabel='count'>
```



most popular payment

```
data["Payment"].value_counts().plot(kind="bar")
```

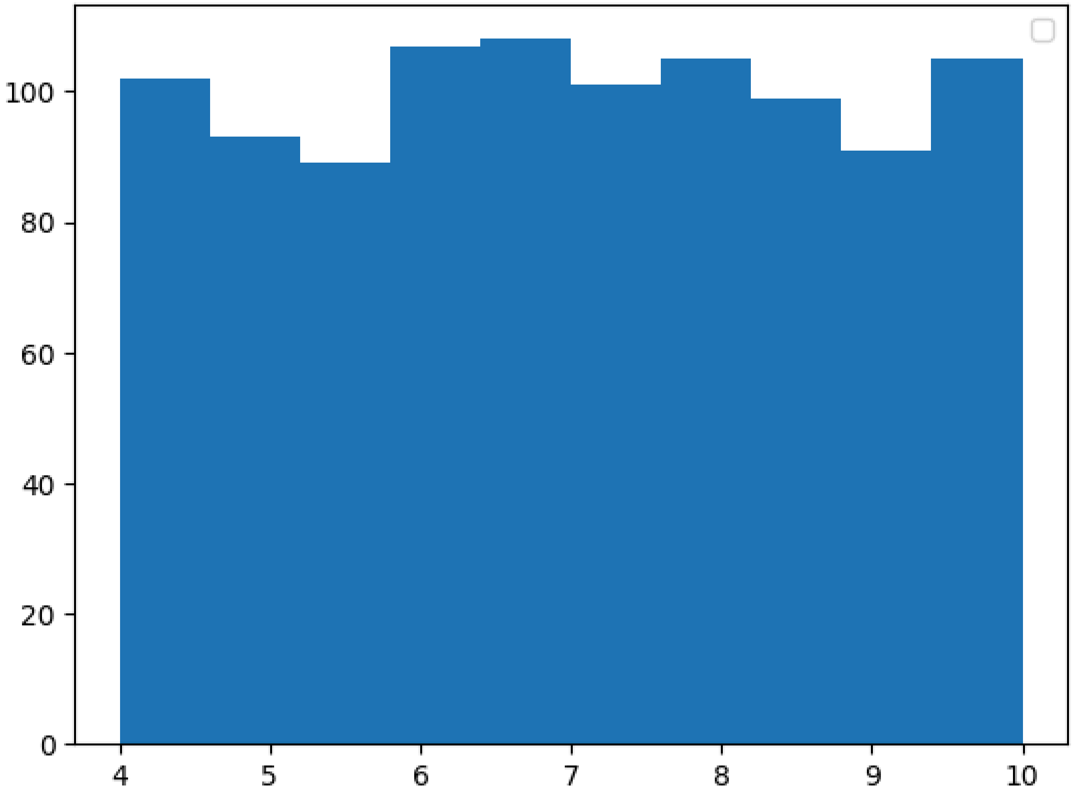
```
<Axes: xlabel='Payment'>
```



distribution of customer ratings

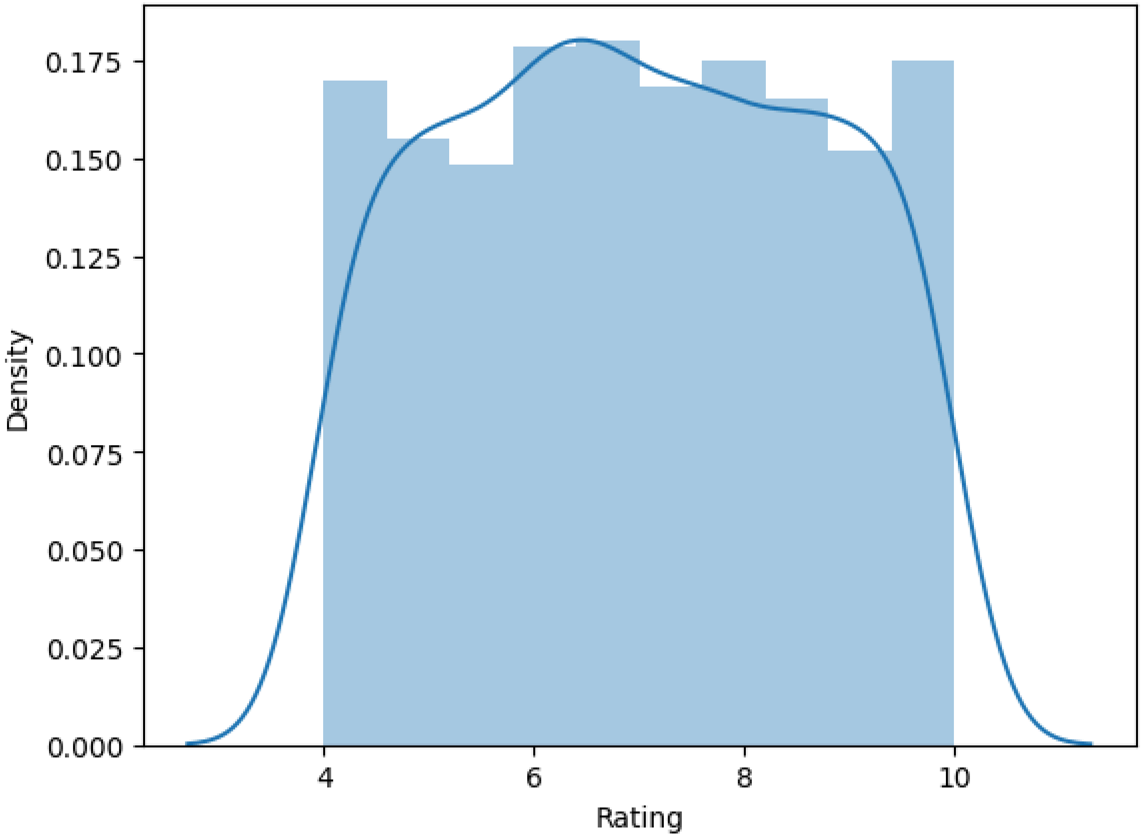
```
plt.hist(data['Rating'])
```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that a
<matplotlib.legend.Legend at 0x7dde0400f460>
```



```
sns.distplot(data["Rating"])
```

```
<Axes: xlabel='Rating', ylabel='Density'>
```



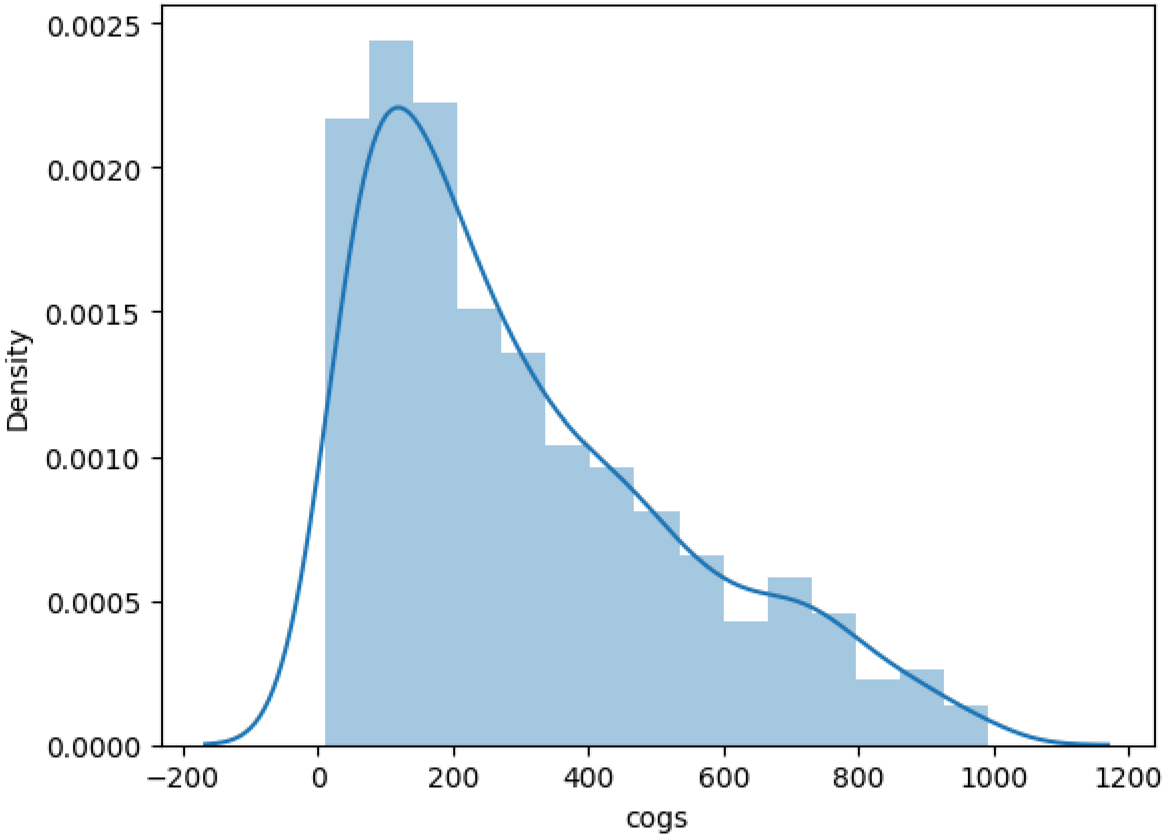
```
data["Rating"].skew()
```

```
0.009009648765730730
```

distribution of cost of goods

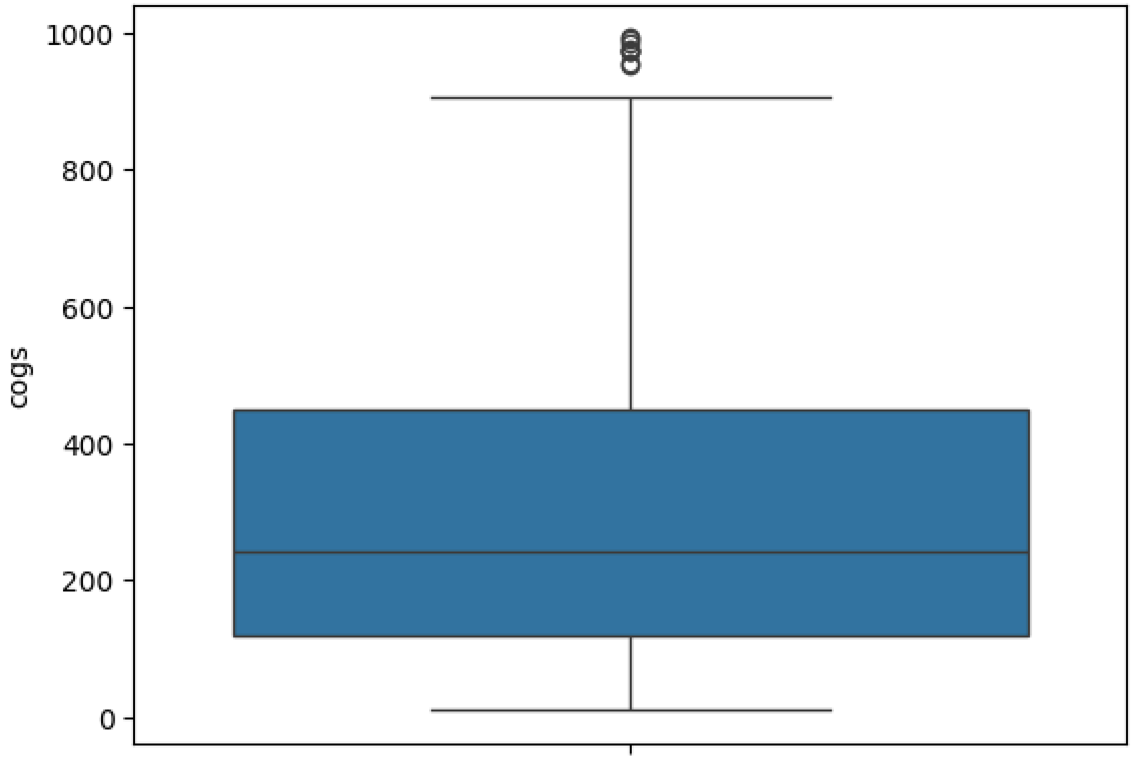
```
sns.distplot(data["cogs"])
```

```
<Axes: xlabel='cogs', ylabel='Density'>
```



```
sns.boxplot(data["cogs"])
```

```
<Axes: ylabel='cogs'>
```

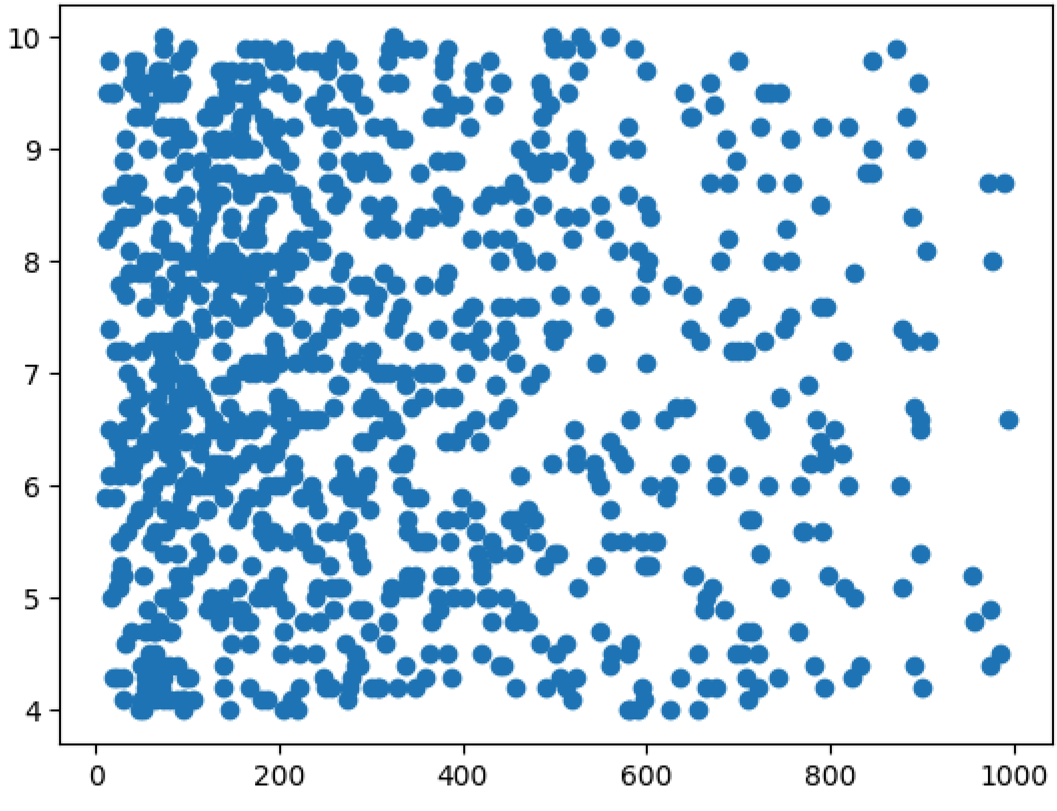


**BIVARIATE ANALYSIS**

cost of goods sold affect the ratings that the customers provide?

```
plt.scatter(data["cogs"],data["Rating"])
```

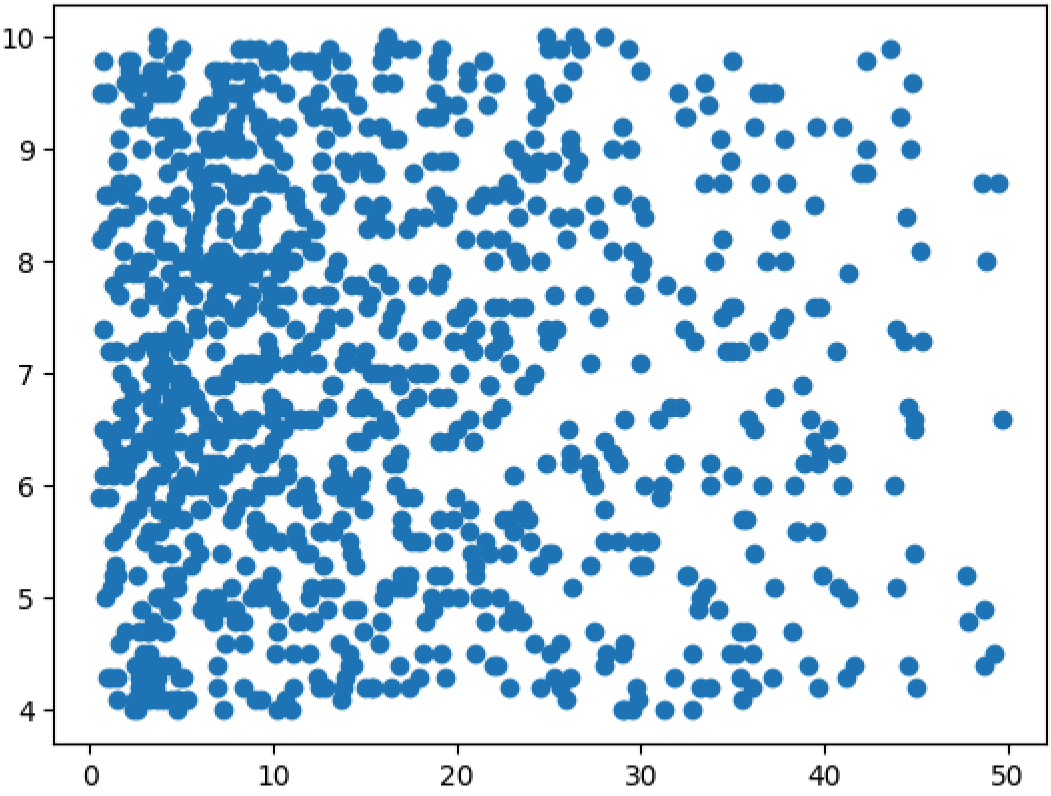
```
<matplotlib.collections.PathCollection at 0x7dde0083c7f0>
```



gross income affect the ratings that the customers provide?

```
plt.scatter(data['gross income'],data['Rating'])
```

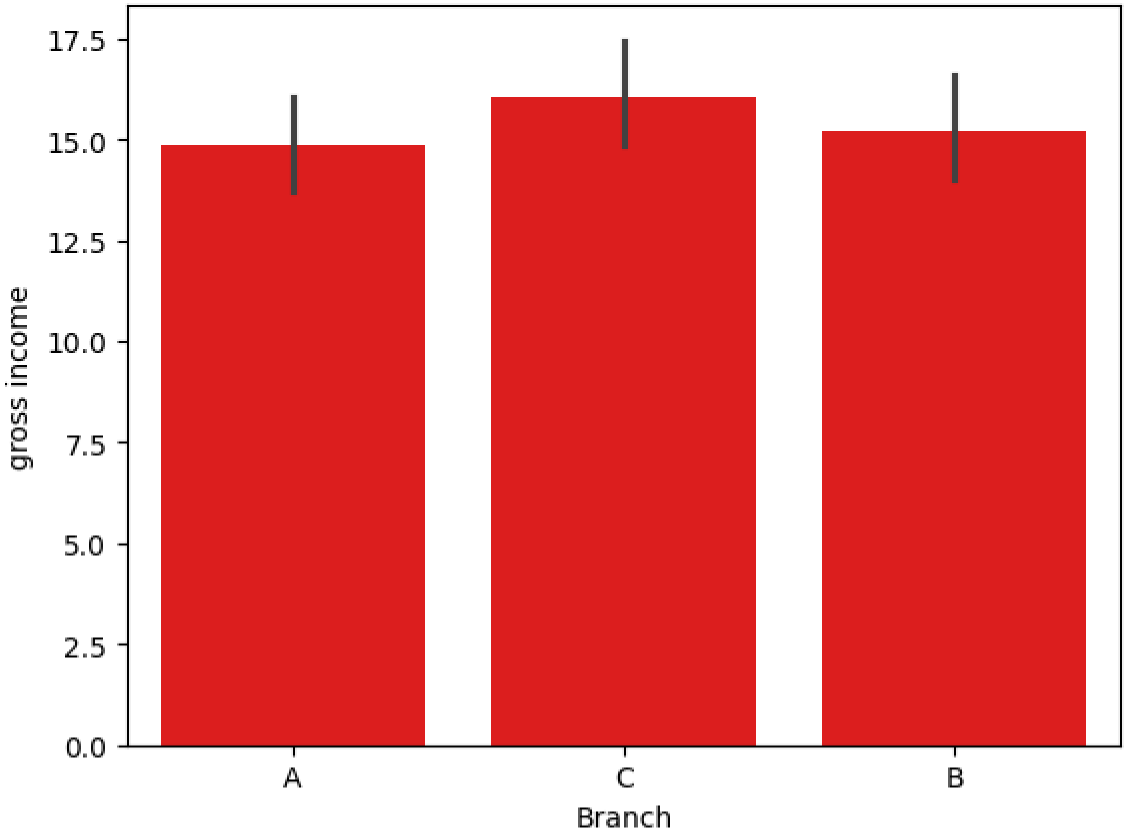
```
<matplotlib.collections.PathCollection at 0x7dde003c5d50>
```



most profitable branch as per gross income

```
sns.barplot(x = data["Branch"],y = data["gross income"],color= "r")
```

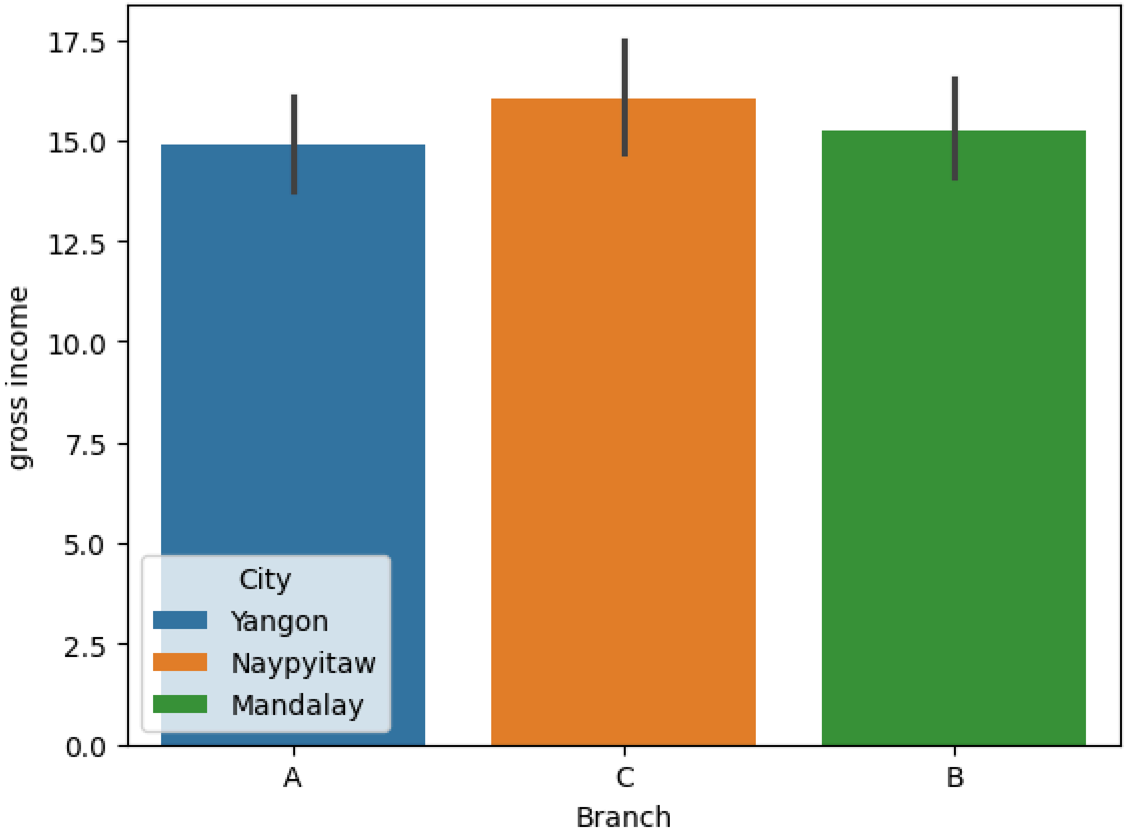
```
<Axes: xlabel='Branch', ylabel='gross income'>
```



### multivariate analysis

```
sns.barplot(x= data['Branch'],y = data['gross income'],hue=data["City"])
```

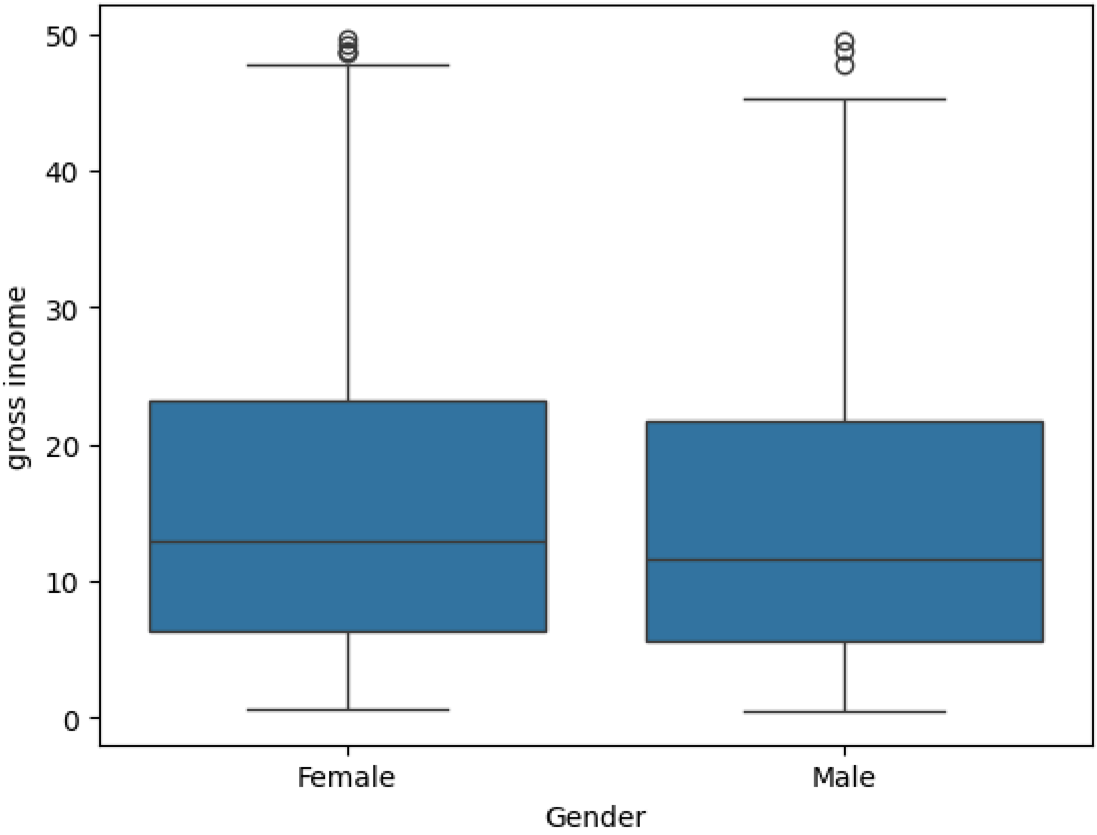
```
<Axes: xlabel='Branch', ylabel='gross income'>
```



Relationship between gender and gross income

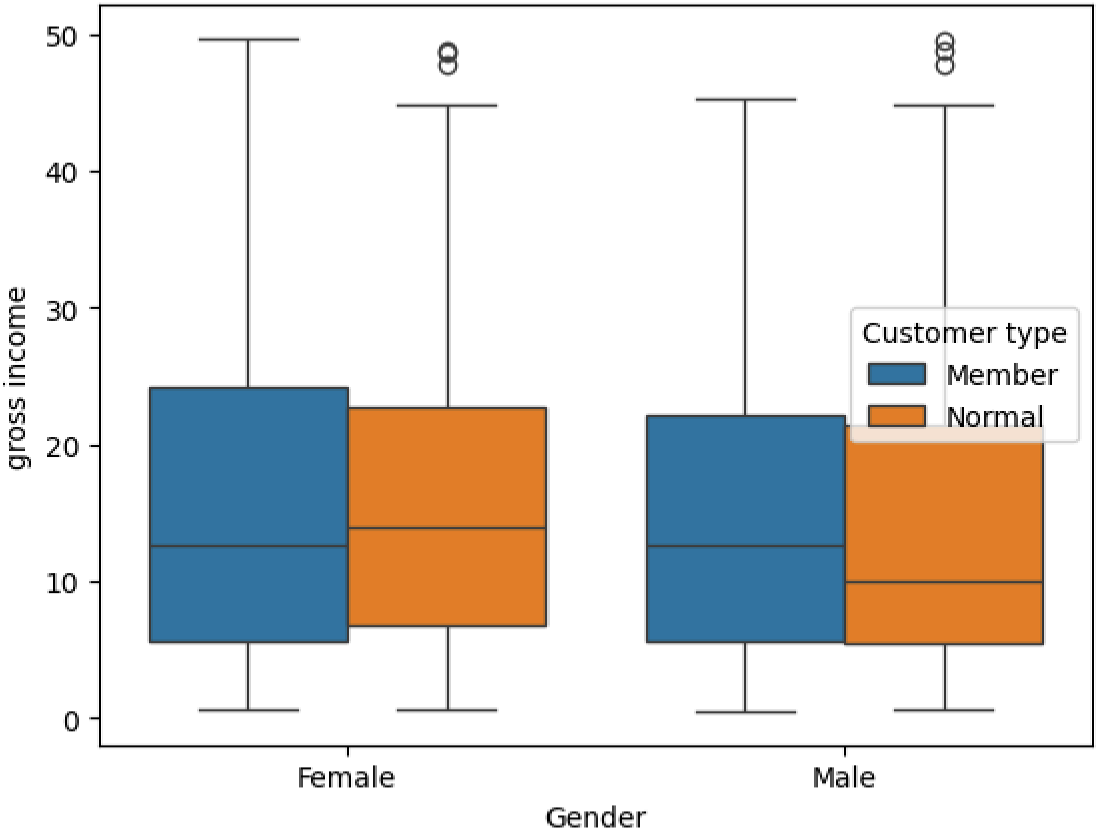
```
sns.boxplot(x=data['Gender'],y=data['gross income'])
```

```
<Axes: xlabel='Gender', ylabel='gross income'>
```



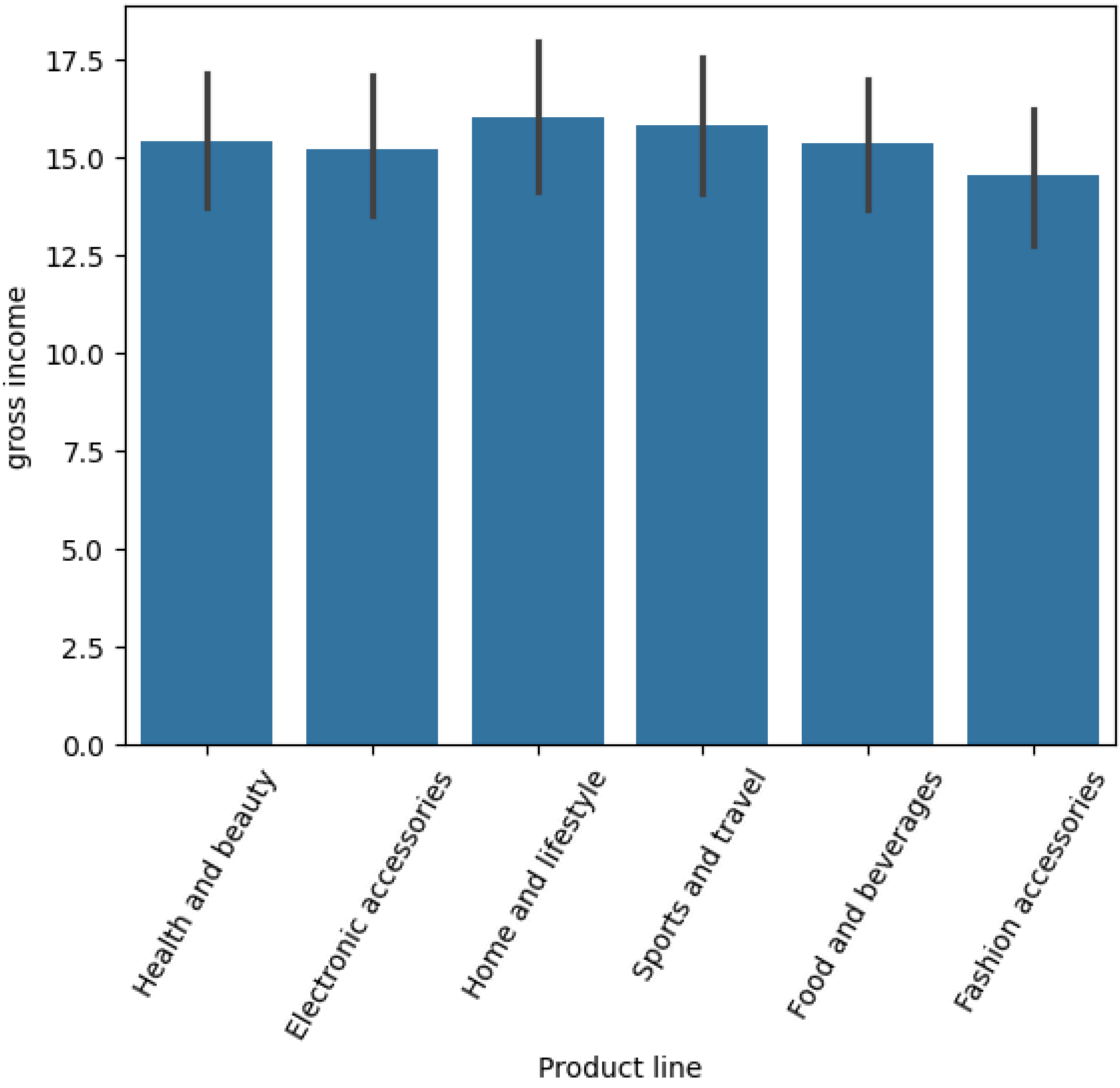
```
sns.boxplot(x=data['Gender'],y=data['gross income'],hue=data['Customer type'])
```

```
<Axes: xlabel='Gender', ylabel='gross income'>
```



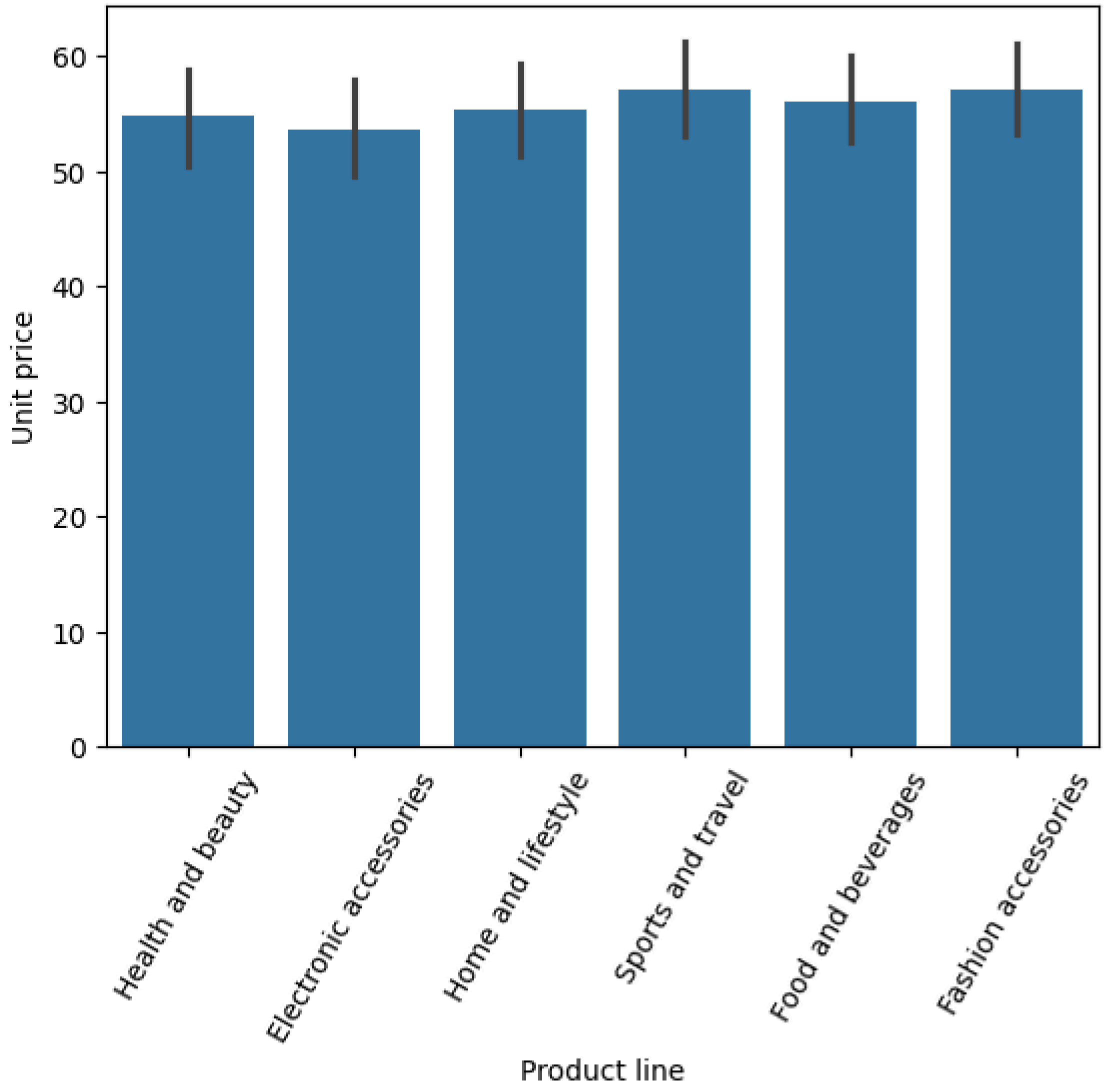
Product line that generates the most income

```
sns.barplot(x = data['Product line'],y= data['gross income'])
plt.xticks(rotation = 60)
plt.show()
```



highest unitprice in the product line

```
sns.barplot(x=data['Product line'],y=data['Unit price'])
plt.xticks(rotation = 60)
plt.show()
```

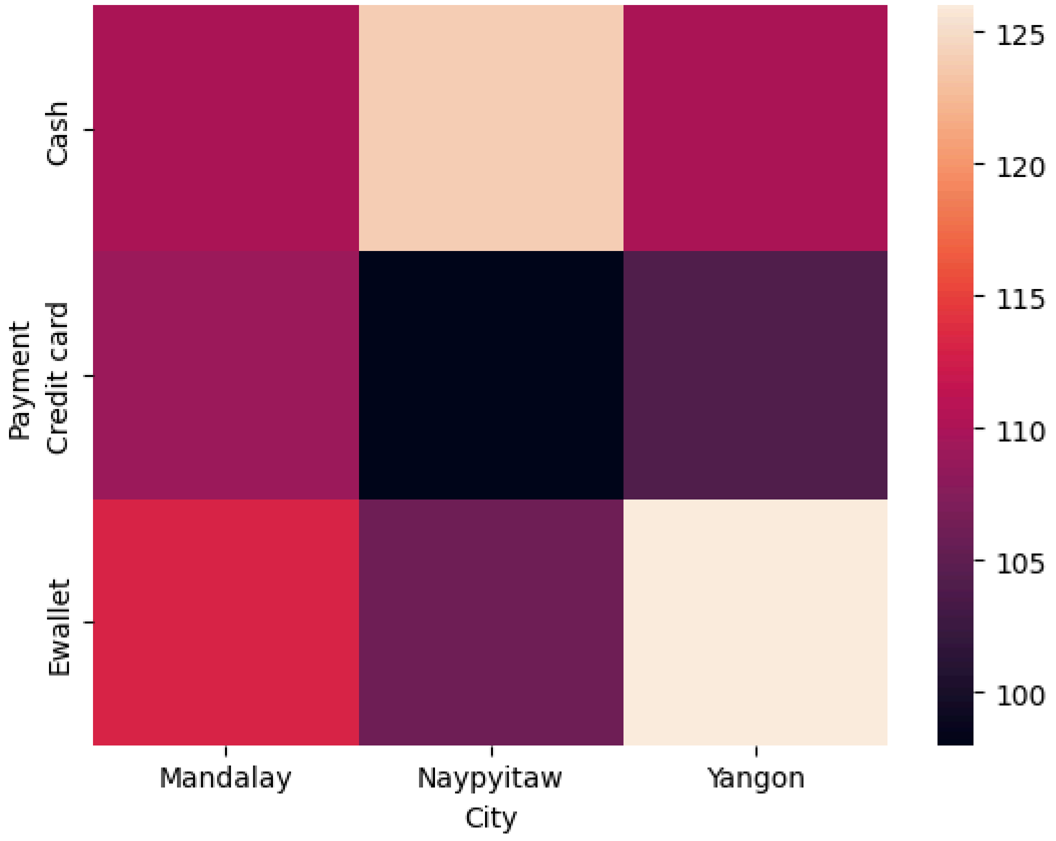


different payment methods used by customers citywise

```
table = pd.crosstab(data['Payment'],data['City'])
```

```
sns.heatmap(table)
```

```
<Axes: xlabel='City', ylabel='Payment'>
```



product line purchased in the highest quantity

```
data['Quantity'].astype(int)
```

|  | Quantity |
|---|---|
| **0** | 7 |
| **1** | 5 |
| **2** | 7 |
| **3** | 8 |
| **4** | 7 |
| **...** | ... |
| **995** | 1 |
| **996** | 10 |
| **997** | 1 |
| **998** | 1 |
| **999** | 7 |

1000 rows × 1 columns

**dtype:** int64

```
grouped = data.groupby('Product line')
data['Product line'].astype(str)
data.groupby('Product line').sum(numeric_only = True)['Quantity']
```