ĐẠI HỌC KINH TẾ TP. HỒ CHÍ MINH

TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ UEH

KHOA CÔNG NGHỆ THÔNG TIN KINH DOANH

ĐỒ ÁN MÔN HỌC

MÔN HỌC: LẬP TRÌNH PHÂN TÍCH DỮ LIỆU

Đề tài nhóm 4:

Phân tích dữ liệu về hành vi mua sắm của khách hàng siêu thị để đề xuất giải pháp nâng cao hiệu quả kinh doanh

STT	HỌ VÀ TÊN	MSSV
1	Đinh Trọng Hữu	31211027643
2	Nguyễn Như Hoàng	31211027640
3	Lê Minh Hoàng	31201024476
4	Lê Ngọc Anh Hùng	31211025877

TP. HỒ CHÍ MINH, NĂM 2

LỜI MỞ ĐẦU

Trong bối cảnh cuộc Cách mạng công nghiệp 4.0, thương mại điện tử đã trở thành xu thế tất yếu và là một trong những kênh phân phối hàng hóa quan trọng. Theo thống kê, doanh thu thương mại điện tử toàn cầu đạt 4.2 nghìn tỷ USD vào năm 2020 và dự kiến đạt 5 nghìn tỷ USD vào năm 2022. Từ đó cho thấy, thị trường thương mại điện tử cũng đang tăng trưởng vượt bậc trong những năm gần đây.

Để duy trì sự tăng trưởng và phát triển bền vững, các doanh nghiệp thương mại điện tử cần xây dựng chiến lược kinh doanh phù hợp, trong đó việc phân tích và dự báo hành vi mua sắm của khách hàng đóng vai trò hết sức quan trọng. Thông qua phân tích các yếu tố ảnh hưởng đến quyết định mua sắm như đặc điểm nhân khẩu học, thói quen tiêu dùng, mức độ tương tác, các doanh nghiệp có thể đánh giá đúng nhu cầu và dự báo chính xác xu hướng tiêu dùng của khách hàng. Từ đó, doanh nghiệp có thể điều chỉnh chiến lược kinh doanh phù hợp để thu hút và gia tăng khả năng mua hàng của khách hàng.

Trong báo cáo này, nhóm sẽ sử dụng bộ dữ liệu “Supermarket Sales” với 1000 quan sát mẫu về lịch sử mua sắm của khách hàng tại siêu thị nhằm mục đích tổng hợp và phân tích các thông tin và các yếu tố ảnh hưởng trong bộ dữ liệu, từ đó đề xuất một số khuyến nghị giúp nâng cao hiệu quả hoạt động kinh doanh của siêu thị.

PHỤ LỤC

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1. Mục tiêu nghiên cứu

Mục tiêu chính của nghiên cứu là phân tích các thông tin liên quan đến hoạt động kinh doanh của siêu thị, bao gồm doanh số bán hàng, các nhóm khách hàng, sản phẩm và hiệu quả hoạt động của từng chi nhánh. Trên cơ sở đánh giá các kết quả phân tích, nghiên cứu đề xuất một số giải pháp và khuyến nghị thiết thực nhằm nâng cao hiệu quả kinh doanh của siêu thị. Cụ thể, nghiên cứu tập trung vào các mục tiêu sau:

- Phân tích diễn biến doanh số bán hàng theo thời gian, sản phẩm, khách hàng để đánh giá đúng thực trạng hoạt động kinh doanh của siêu thị.
- Đánh giá sự khác biệt về hiệu quả hoạt động kinh doanh giữa các chi nhánh.
- Xác định các nhóm khách hàng tiềm năng và sản phẩm có tiềm năng phát triển của siêu thị
- Đề xuất các giải pháp marketing cũng như chiến lược kinh doanh phù hợp giúp nâng cao hiệu quả hoạt động siêu thị.

Những mục tiêu này hướng tới mục đích cuối cùng là nâng cao năng lực cạnh tranh và hiệu quả kinh doanh lâu dài cho siêu thị.

2. Phương pháp nghiên cứu

- Thăm dò, phân tích dữ liệu.
- Trực quan hóa thông tin từ tập dữ liệu.
- Tiến hành làm sạch dữ liệu (loại bỏ nhiễu,outlier, giữ lại thông tin quan trọng).
- Phân tích đơn biến.
- Kiểm định giả thuyết ANOVA 1 chiều phân tích đa biến (hai biến, ba biến).
- Xây dựng mô hình.Đánh giá mô hình.

3. Tài liệu sử dụng

- Ngôn ngữ lập trình Python.
- Bộ dữ liệu “supermarket sales” trong Kaggle.

CHƯƠNG 2: TỔNG QUAN VỀ BỘ DỮ LIỆU

1. Mô tả bộ dữ liệu được cung cấp:

Bộ dữ liệu "Supermarket sales" chứa các thông tin về lịch sử doanh số bán hàng của một công ty siêu thị đã được ghi nhận ở ba chính nhánh khác nhau trong ba tháng đầu của năm 2019 với 1000 mẫu. Bộ dữ liệu bao gồm có 17 thuộc tính.

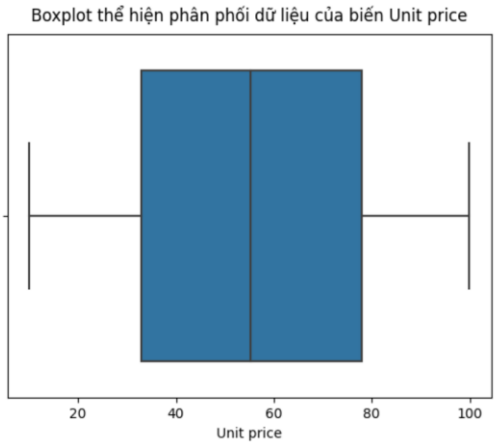
2. Các thuộc tính của bộ dữ liệu:

2.1. Unit price:

Input[2.1]:

```
# Các thông số xác suất đánh giá thuộc tính
print('Các thông số xác suất của biến \'Unit price\':\n')
print(df['Unit price'].describe())
```

Outpu[2.1]:




```
Các thông số xác suất của biến 'Unit price':

count    1000.000000
mean       55.672130
std        26.494628
min        10.080000
25%        32.875000
50%        55.230000
75%        77.935000
max        99.960000
Name: Unit price, dtype: float64
```

Hình2.1. Thông tin mô tả về biến Unit price.

Nhận xét:

- Biến Unit price ghi nhận thông tin về giá của mỗi sản phẩm (đơn vị: USD)
- Giá của các sản phẩm được ghi nhận nằm trong khoảng [10.08, 99.96] USD.
- Giá trung bình của các sản phẩm khoảng 55.67 USD.
- Độ lệch chuẩn của giá các sản phẩm là khoảng 26.5.

Kiểu dữ liệu: Numerical.

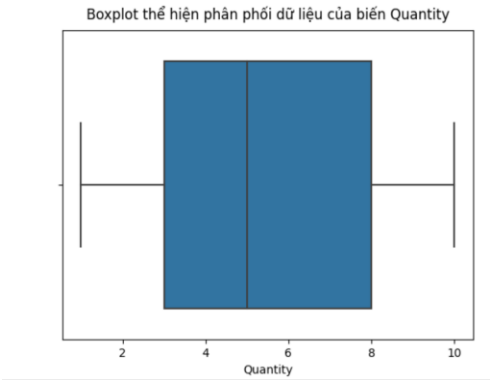
2.2. Quantity:

Input[2.2]:

```
# Các thông số xác suất đánh giá thuộc tính
print('Các thông số xác suất của biến \'Quantity\':\n')
```

```
print(df['Quantity'].describe())
```

Output[2.2]:




```
Các thông số xác suất của biến 'Quantity':

count    1000.000000
mean        5.510000
std         2.923431
min         1.000000
25%         3.000000
50%         5.000000
75%         8.000000
max        10.000000
Name: Quantity, dtype: float64
```

Hình2.2.Thông tin mô tả về biến Quantity.

Nhận xét:

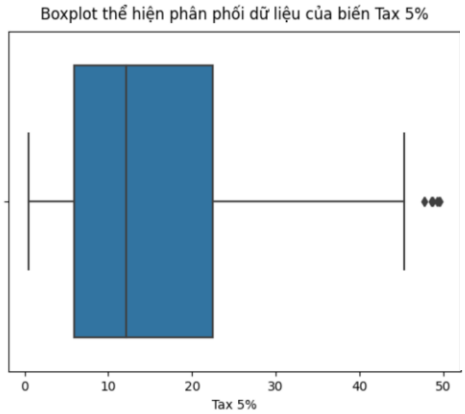
- Biến Quantity ghi nhận thông tin về số lượng của mỗi sản phẩm.
- Số lượng mỗi sản phẩm được ghi nhận nằm trong khoảng từ [1,10] sản phẩm.
- Số lượng trung bình là khoảng 5 sản phẩm.
- Độ lệch chuẩn là khoảng 3 sản phẩm.

2.3. Tax 5%:

Input[2.3]:

```
# Các thông số xác suất đánh giá thuộc tính
print('Các thông số xác suất của biến \'Tax 5%\':\n')
print(df['Tax 5%'].describe())
```

Output[2.3]:




```
Các thông số xác suất của biến 'Tax 5%':

count    1000.000000
mean       15.379369
std        11.708825
min         0.508500
25%         5.924875
50%        12.088000
75%        22.445250
max        49.650000
Name: Tax 5%, dtype: float64
```

Hình 2.3. Thông tin mô tả về biến Tax 5%.

Nhận xét:

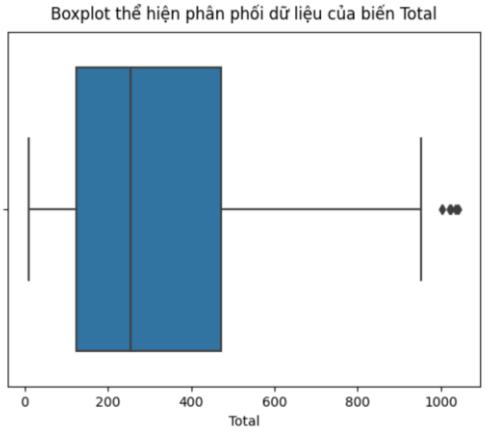
- Biến Tax 5% ghi nhận thông tin về mức thuế 5% tính vào tổng hóa đơn của khách hàng khi mua hàng.
- Thuế được ghi nhận nằm trong khoảng [0.51,49.65].
- Thuế trung bình khoảng 15.38.
- Độ lệch chuẩn khoảng 11.71.

2.4. Total:

Input[2.4]:

```
# Các thông số xác suất đánh giá thuộc tính
print('Các thông số xác suất của biến \'Total\':\n')
print(df['Total'].describe())
```

Output[2.4]:




```
Các thông số xác suất của biến 'Total':

count    1000.000000
mean      322.966749
std       245.885335
min        10.678500
25%       124.422375
50%       253.848000
75%       471.350250
max      1042.650000
Name: Total, dtype: float64
```

Hình 2.4.Thông tin mô tả về biến Total.

Nhận xét:

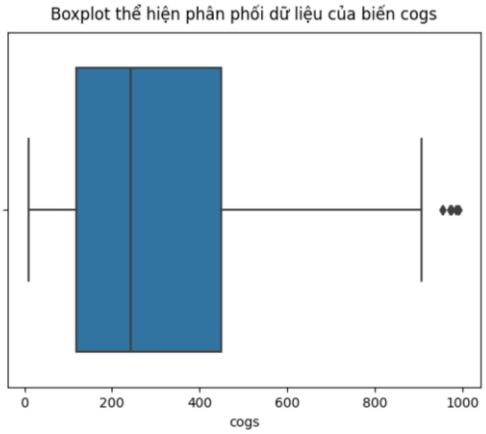
- Biến Total ghi nhận thông tin về tổng chi tiêu của khách hàng cho 1 hóa đơn.
- Biến Total được ghi nhận nằm trong khoảng [10.68,1042.65].
- Tổng chi tiêu trung bình khoảng 322.97.
- Độ lệch chuẩn khoảng 245.89.

2.5. Cogs:

Input[2.5]:

```
# Các thông số xác suất đánh giá thuộc tính
print('Các thông số xác suất của biến \'cogs\':\n')
print(df['cogs'].describe())
```

Output[2.5]:




```
Các thông số xác suất của biến 'cogs':

count    1000.00000
mean      307.58738
std       234.17651
min        10.17000
25%       118.49750
50%       241.76000
75%       448.90500
max       993.00000
Name: cogs, dtype: float64
```

Hình 2.5.Thông tin mô tả về biến cogs.

Nhận xét:

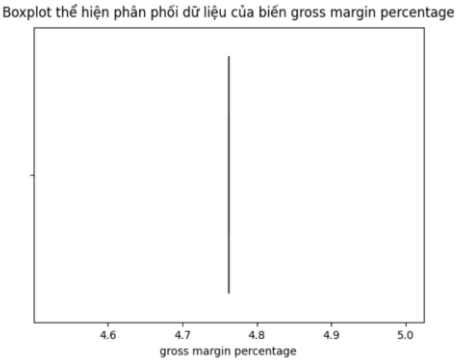
- Biến cogs ghi nhận thông tin về giá vốn hàng bán của sản phẩm.
- Biến cogs được ghi nhận nằm trong khoảng [10.17,993].
- cogs trung bình khoảng 307.59.
- Độ lệch chuẩn khoảng 234.18.

2.6. Gross margin percentage:

Input[2.6]:

```
# Các thông số xác suất đánh giá thuộc tính
print('Các thông số xác suất của biến \'gross margin percentage\':\n')
print(df['gross margin percentage'].describe())
```

Output[2.6]:




```
Các thông số xác suất của biến 'gross margin percentage':

count    1000.000000
mean        4.761905
std         0.000000
min         4.761905
25%         4.761905
50%         4.761905
75%         4.761905
max         4.761905
Name: gross margin percentage, dtype: float64
```

Hình2.6.Thông tin mô tả về biến gross margin percentage.

Nhận xét:

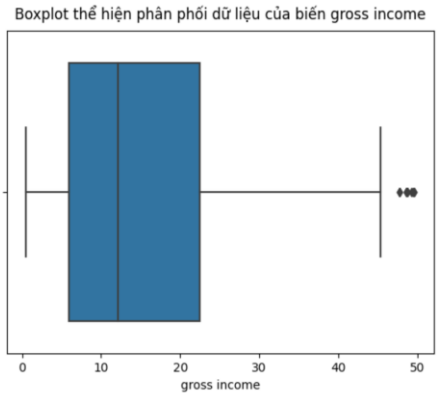
- Biến gross margin percentage ghi nhận thông tin về tỉ suất lợi nhuận gộp, vì đây là một chỉ số đo lường cố định được tính bằng chia lợi nhuận gộp cho doanh thu rồi nhân 100% để ra tỉ lệ phần trăm, do đó chỉ ghi nhận được một giá trị duy nhất.

2.7. Gross income:

Input[2.7]:

```
# Các thông số xác suất đánh giá thuộc tính
print('Các thông số xác suất của biến \'gross income\':\n')
print(df['gross income'].describe())
```

Output[2.7]:




```
Các thông số xác suất của biến 'gross income':

count    1000.000000
mean       15.379369
std        11.708825
min         0.508500
25%         5.924875
50%        12.088000
75%        22.445250
max        49.650000
Name: gross income, dtype: float64
```

Hình2.7.Thông tin mô tả về biến Gross income.

Nhận xét:

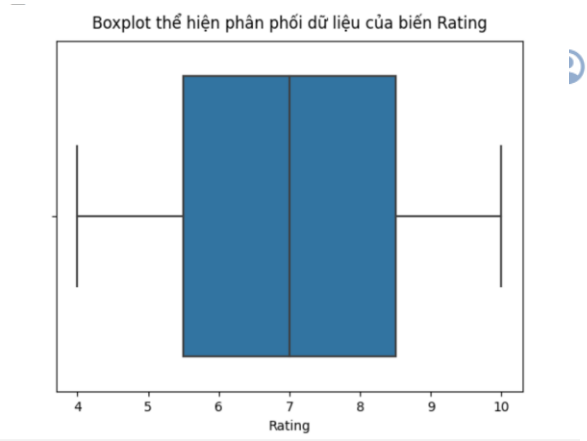
- Biến gross income ghi nhận thông tin về tổng doanh thu.
- Biến gross income được ghi nhận nằm trong khoảng [0.51,49.65]
- Gross income trung bình khoảng 15.38.
- Độ lệch chuẩn khoảng 11.71.

2.8. Rating:

Input:

```
# Các thông số xác suất đánh giá thuộc tính
print('Các thông số xác suất của biến \'Rating\':\n')
print(df['Rating'].describe())
```

Output:




```
Các thông số xác suất của biến 'Rating':

count    1000.00000
mean        6.97270
std         1.71858
min         4.00000
25%         5.50000
50%         7.00000
75%         8.50000
max        10.00000
Name: Rating, dtype: float64
```

Hình 2.8.Thông tin mô tả về biến Rating.

Nhận xét:

- Biến Rating ghi nhận thông tin về đánh giá xếp loại trải nghiệm của khách hàng khi đi mua sắm.
- Biến Rating được ghi nhận nằm trong khoảng [4,10].
- Rating trung bình khoảng 7.

Độ lệch chuẩn khoảng 1.72.

2.9. Branch:

Input:

```
# Các thông số quan sát thuộc tính
print('Các thông số xác suất của biến \'Branch\':')
print(df.Branch.describe())
print('Các giá trị unique của biến \'Branch\' cụ thể là:',df.Branch.unique())
```

Output:

```
Các thông số xác suất của biến 'Branch':
count     1000
unique       3
top          A
freq       340
Name: Branch, dtype: object
Các giá trị unique của biến 'Branch' cụ thể là: ['A' 'C' 'B']
```

Hình 2.9.Thông tin mô tả về biến Branch.

Nhận xét:

- Biến Branch ghi nhận thông tin về 3 chi nhánh của công ty siêu thị, 3 chi nhánh đó là:
  + A
  + B

+ C

- Chi nhánh được ghi nhận nhiều nhất là chi nhánh A số lần ghi nhận là 340 lần.

2.10. City:

Input:

```
# Các thông số quan sát thuộc tính
print('Các thông số xác suất của biến \'City\':')
print(df.City.describe())
print('Các giá trị unique của biến \'City\' cụ thể là:',df.City.unique())
```

Output:

```
Các thông số xác suất của biến 'City':
count       1000
unique         3
top       Yangon
freq         340
Name: City, dtype: object
Các giá trị unique của biến 'City' cụ thể là: ['Yangon' 'Naypyitaw' 'Mandalay']
```

Hình 2.10.Thông tin mô tả về biến City.

Nhận xét:

- Biến City ghi nhận thông tin về 3 thành phố nơi 3 ba chi nhánh của công ty siêu thị được đặt. Ba thành phố đó là:
  + Yangon
  + Naypyitaw
  + Mandalay
- Tương tự như chi nhánh thì thành phố được ghi nhận nhiều nhất đó là thành phố Yangon với số lần ghi nhận là 340.

2.11. Customer type:

Input:

```
# Các thông số quan sát thuộc tính
print('Các thông số xác suất của biến \'Customer type\':')
print(df['Customer type'].describe())
print('Các giá trị unique của biến \'Customer type\' cụ thể là:',df['Customer
type'].unique())
```

Output:

```
Các thông số xác suất của biến 'Customer type':
count       1000
unique         2
top       Member
freq         501
Name: Customer type, dtype: object
Các giá trị unique của biến 'Customer type' cụ thể là: ['Member' 'Normal']
```

Hình 2.11. Thông tin mô tả về biến Customer type.

Nhận xét:

- Biến Customer type ghi nhận thông tin về phân loại khách hàng.Có 2 loại khách hàng đó là Member và Normal, trong đó:
  + Member là những người mua hàng quyết định đăng ký thẻ thành viên tại các chi nhánh của công ty siêu thị.
  + Normal là những khách hàng không chọn đăng ký trở thành thành viên của các chi nhánh.
- Tệp khách hàng được ghi nhận nhiều hơn đó là Member với số lượng là 501 trong khi đó Normal với số lượng là 499.

2.12. Gender:

Input:

```
# Các thông số quan sát thuộc tính
print('Các thông số xác suất của biến \'Gender\':')
print(df.Gender.describe())
print('Các giá trị unique của biến \'Gender\' cụ thể là:',df.Gender.unique())
```

Output:

```
Các thông số xác suất của biến 'Gender':
count       1000
unique         2
top       Female
freq         501
Name: Gender, dtype: object
Các giá trị unique của biến 'Gender' cụ thể là: ['Female' 'Male']
```

Hình 2.12. Thông tin mô tả về biến Gender.

Nhận xét:

- Biến Gender ghi nhận thông tin về giới tính khách hàng đó là Male và Female.
- Female được ghi nhận nhiều hơn 501 trong khi Male là 499.

2.13. Product line:

Input:

```
# Các thông số quan sát thuộc tính
print('Các thông số xác suất của biến \'Product_line\':')
print(df['Product line'].describe())
print('Các giá trị unique của biến \'Product_line\' cụ thể là:',df['Product
line'].unique())
```

Output:

```
Các thông số xác suất của biến 'Product_line':
count                  1000
unique                    6
top       Fashion accessories
freq                    178
Name: Product line, dtype: object
Các giá trị unique của biến 'Product_line' cụ thể là: ['Health and beauty' 'Electronic accessories' 'Home and lifestyle'
 'Sports and travel' 'Food and beverages' 'Fashion accessories']
```

Hình 2.13.Thông tin mô tả về biến Product line.

Nhận xét:

- Biến Product line ghi nhận thông tin về số lượng các dòng sản phẩm tại các chi nhánh siêu thị. Có tất cả 6 loại sản phẩm:
  + Health and beauty: Sản phẩm chăm sóc sức khỏe và làm đẹp.
  + Electronic accessories: Sản phẩm về điện.
  + Home and lifestyle: Đồ gia dụng.
  + Sports and travel: Thể thao và du lịch.
  + Food and beverages: đồ ăn và thức uống.
  + Fashion accessories: thời trang và phụ kiện.
- Trong đó, thời trang và phụ kiện là mặt hàng được bán chạy nhất với 178 sản phẩm được bán ra.

2.14. Payment:

Input:

```
# Các thông số quan sát thuộc tính
print('Các thông số xác suất của biến \'Payment\':')
print(df.Payment.describe())
print('Các giá trị unique của biến \'Payment\' cụ thể là:',df.Payment.unique())
```

Output:

```
Các thông số xác suất của biến 'Payment':
count        1000
unique          3
top       Ewallet
freq          345
Name: Payment, dtype: object
Các giá trị unique của biến 'Payment' cụ thể là: ['Ewallet' 'Cash' 'Credit card']
```

Hình 2.14. Thông tin mô tả về biến Payment.

Nhận xét:

- Biến Payment ghi nhận thông tin về các phương thức thường được sử dụng khi đi mua sắm của khách hàng tại các cửa hàng.
- Có 3 loại phương thức chính đó là:
    + Ewallet: ví điện tử.
    + Cash: tiền mặt.
    + Credit card: thẻ tín dụng.
- Trong đó, ví điện thử là phương thức thanh toán được sử dụng nhiều nhiều nhất với 345 lần sử dụng.

2.15. Date:

Input:

```
# Các thông số quan sát thuộc tính
print('Các thông số xác suất của biến \'Date\':')
print(df.Date.describe())
print('Các giá trị unique của biến \'Date\' cụ thể là:',df.Date.unique())
```

Output:

```
Các thông số xác suất của biến 'Date':
count         1000
unique          89
top       2/7/2019
freq            20
Name: Date, dtype: object
Các giá trị unique của biến 'Date' cụ thể là: ['1/5/2019' '3/8/2019' '3/3/2019' '1/27/2019' '2/8/2019' '3/25/2019'
 '2/25/2019' '2/24/2019' '1/10/2019' '2/20/2019' '2/6/2019' '3/9/2019'
 '2/12/2019' '2/7/2019' '3/29/2019' '1/15/2019' '3/11/2019' '1/1/2019'
 '1/21/2019' '3/5/2019' '3/15/2019' '2/17/2019' '3/2/2019' '3/22/2019'
 '3/10/2019' '1/25/2019' '1/28/2019' '1/7/2019' '3/23/2019' '1/17/2019'
 '2/2/2019' '3/4/2019' '3/16/2019' '2/27/2019' '2/10/2019' '3/19/2019'
 '2/3/2019' '3/7/2019' '2/28/2019' '3/27/2019' '1/20/2019' '3/12/2019'
 '2/15/2019' '3/6/2019' '2/14/2019' '3/13/2019' '1/24/2019' '1/6/2019'
 '2/11/2019' '1/22/2019' '1/13/2019' '1/9/2019' '1/12/2019' '1/26/2019'
 '1/23/2019' '2/23/2019' '1/2/2019' '2/9/2019' '3/26/2019' '3/1/2019'
 '2/1/2019' '3/28/2019' '3/24/2019' '2/5/2019' '1/19/2019' '1/16/2019'
 '1/8/2019' '2/18/2019' '1/18/2019' '2/16/2019' '2/22/2019' '1/29/2019'
 '1/4/2019' '3/30/2019' '1/30/2019' '1/3/2019' '3/21/2019' '2/13/2019'
 '1/14/2019' '3/18/2019' '3/20/2019' '2/21/2019' '1/31/2019' '1/11/2019'
 '2/26/2019' '3/17/2019' '3/14/2019' '2/4/2019' '2/19/2019']
```

Hình 2.15. Thông tin mô tả về biến Date.

Nhận xét:

- Biến Date ghi nhận thông tin về ngày, tháng và năm của các giao dịch diễn ra tại ba chính nhánh.

2.16. Time:

Input:

```
# Các thông số quan sát thuộc tính
```

```
print('Các thông số xác suất của biến \'Time\':')
print(df.Time.describe())
print('Các giá trị unique của biến \'Time\' cụ thể là:',df.Time.unique())
```

Output:

```
Các giá trị unique của biến 'Time' cụ thể là: ['13:08' '10:29' '13:23' '20:33' '10:37' '18:30' '14:36' '11:38' '17:15'
 '13:27' '18:07' '17:03' '10:25' '16:48' '19:21' '16:19' '11:03' '10:39'
 '18:00' '15:30' '11:24' '10:40' '12:20' '11:15' '17:36' '19:20' '15:31'
 '12:17' '19:48' '15:36' '19:39' '12:43' '14:49' '10:12' '10:42' '12:28'
 '19:15' '17:17' '13:24' '13:01' '18:45' '10:11' '13:03' '20:39' '19:47'
 '17:24' '15:47' '12:45' '17:08' '10:19' '15:10' '14:42' '15:46' '11:49'
 '19:01' '11:26' '11:28' '15:55' '20:36' '17:47' '10:55' '13:40' '12:27'
 '14:35' '16:40' '15:43' '15:01' '10:04' '18:50' '12:46' '18:17' '18:21'
 '17:04' '14:20' '15:48' '16:24' '18:56' '19:56' '18:37' '10:17' '14:31'
 '10:23' '20:35' '16:57' '17:55' '19:54' '16:42' '12:09' '20:05' '20:38'
 '13:11' '10:16' '18:14' '13:22' '11:27' '16:44' '18:19' '14:50' '20:54'
 '20:19' '10:43' '14:30' '11:32' '10:41' '12:44' '20:07' '20:31' '12:29'
 '15:26' '20:48' '12:02' '17:26' '19:52' '14:57' '18:44' '13:26' '16:17'
```

Hình 2.16. Thông tin mô tả về biến Time.

Nhận xét:

- Biến Time ghi nhận thông tin về thời gian, phút, giờ mà các giao dịch diễn ra.

3. Xây dựng hàm bootstrap cho các biến số:

- Sơ lược về phương pháp bootstrap:

  + Phương pháp Bootstrapping do nhà thống kê học Bradley Efron thuộc đại học Stanford (Mỹ) phát triển từ cuối thập niên 1979s nhưng đến khi máy tính được sử dụng phổ biến thì phương pháp này mới trở thành phương pháp phổ biến trong phân tích thống kê và được ứng dụng rộng rãi trong rất nhiều lĩnh vực khoa học.

  + Bootstrap method là phương pháp lấy mẫu có hoàn lại (sampling with replacement). Phương pháp lấy mẫu có hoàn lại có nghĩa là một cá thể có thể xuất hiện nhiều lần trong một lần lấy mẫu.

  + Tại sao cần Bootstrap method? Trong thực tế, từ một mẫu ta chỉ có thể có được một số trung bình của mẫu, ta không biết được khoảng tin cậy cho số trung bình này hoặc không biết được phân bố của số trung bình ra sao. Thêm vào đó thực tế ta không biết được hàm phân bố của tổng thể nên việc ước lượng các tham số đặc trưng thống kê rất khó khăn và thiếu chính xác. Bootstrap method có thể cung cấp nhiều thông tin chi tiết hơn hơn về phân bố của số trung bình, khoảng tin cậy cũng như xác suất của số trung bình dựa trên một mẫu duy nhất.

  + Các bước chính của Bootstrap method:

  Sinh ra các mẫu (Bootstrap sampling) ngẫu nhiên có hoàn lại kích thước n từ tổng thể (từ mẫu ban đầu).

Tính các thông số thống kê đặc trưng cho của mẫu được sinh ra (mean, Confident interval, Standard Deviation, Inter Quartile,…).

Lặp lại bước 1 và bước 2 với số lần lớn (thường trên 1000).

Sử dụng các ước lượng thống kê của Bootstrap sampling đã tính ở bước 2 để đánh giá độ chính xác các ước lượng thống kê của mẫu ban đầu (Original sample, Training Data).

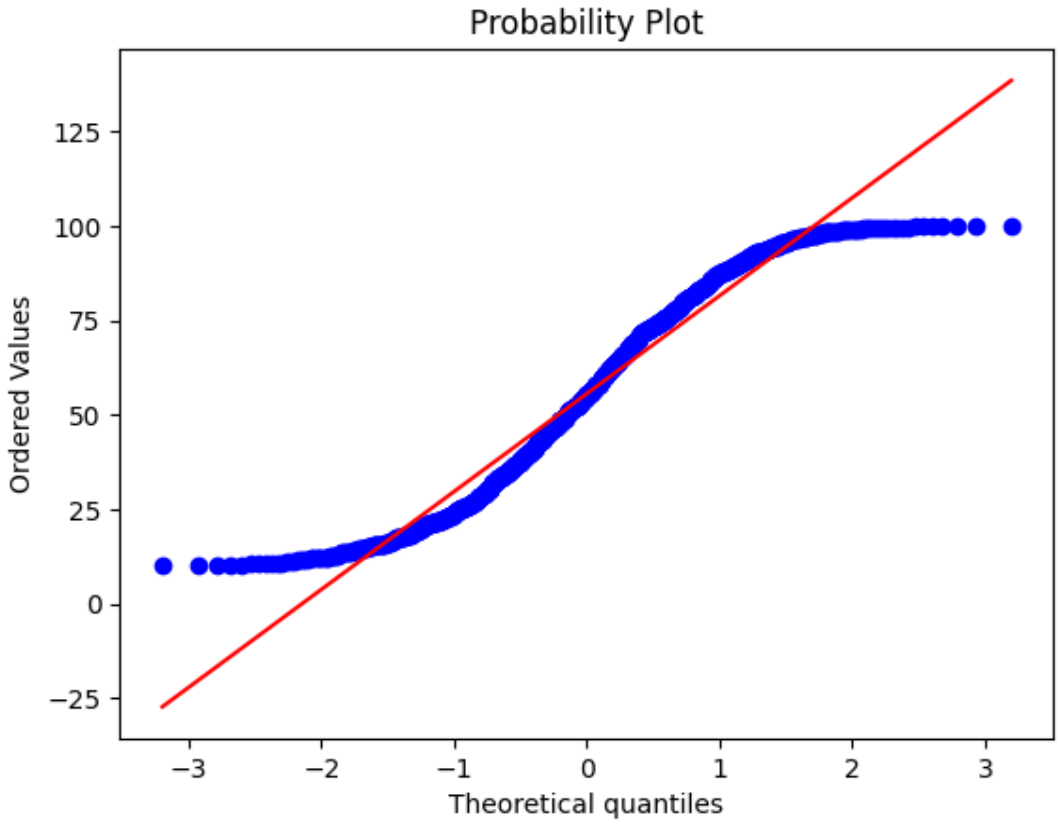
- Xây dựng hàm:

```
#Xây dựng hàm để ước lượng giá trị kỳ vọng, phương sai, percentiles, và các thống kê khác
def bootstrap_fun(sample, fn, m, alpha):

    # Lấy giá trị trung bình, độ lệch chuẩn và kích thước mẫu gốc
    sample_mean = sample.mean()
    sample_std = sample.std()
    n = len(sample)

    # Danh sách phân phối Bootstrap (trung bình)
    bootstrap_dist = []

    # Vòng lặp để lấy giá trị ngẫu nhiên từ mẫu với sự thay thế, tính giá trị trung bình và thêm vào danh sách
    for i in range(m):
        replacement = np.random.choice(sample, n, replace=True)
        estimator = fn(replacement)
        bootstrap_dist.append(estimator)

    # Tính giá trị trung bình, độ lệch chuẩn và sai số sau khi thực hiện Bootstrap
    bootstrap_mean = fn(bootstrap_dist)
    bootstrap_se = np.std(bootstrap_dist)
    bias = bootstrap_mean - sample_mean

    # Tính khoảng tin cậy
    percentile_high = 100 * (1 - alpha / 2)
    percentile_low = 100 * (alpha / 2)

    high = np.percentile(bootstrap_dist, percentile_high)
    median = np.percentile(bootstrap_dist, 50)
    low = np.percentile(bootstrap_dist, percentile_low)

    # In kết quả
    print(f'Mean mẫu gốc: {sample_mean}, với độ lệch chuẩn: {sample_std}')
    print('----')
```

```
    print(f'Mean sau khi thực hiện Bootstrap: {bootstrap_mean}, với độ lệch chuẩn:
{bootstrap_se}')
    print('----')
    print(f'Sai số: {bias}')
    print('----')
    print(f'Khoảng tin cậy {1-alpha:.0%}:\n [{low:.3f} .. {median:.3f} ..
{high:.3f}]')

    #Hiển thị kết quả
    fig, ax = plt.subplots(figsize=(14,7))
    sns.histplot(bootstrap_dist, bins =50, color = '#a0deaa', linewidth =
0.5, edgecolor = 'gray', stat = 'density')
    sns.kdeplot(bootstrap_dist, color = 'black')

    ax.grid(axis = 'y', alpha = 0.5)
    plt.axvline(bootstrap_mean, color = 'red')
    plt.axvline(high, color = 'blue')
    plt.axvline(low, color = 'blue')
    plt.plot()
```

3.1. Unit price:

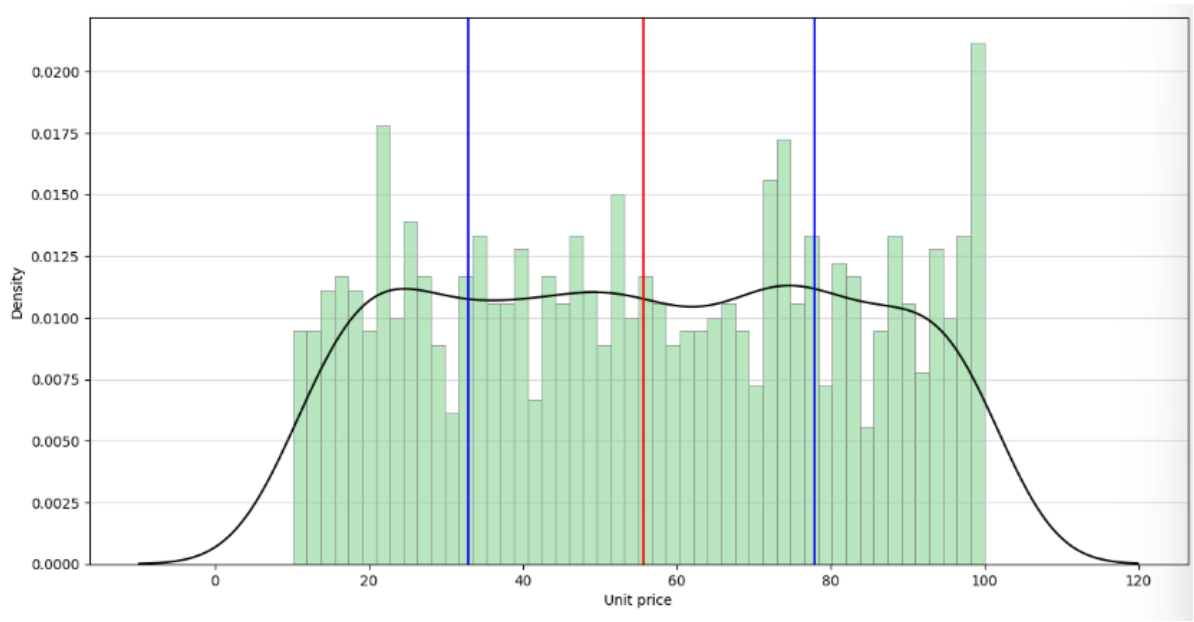
- Kiểm tra độ lệch của dữ liệu với biểu đồ qq-plot:



Hình 2.17 Biểu đồ qq-plot của biến Unit price.

Nhận xét:

- Trong biểu đồ này, các điểm dữ liệu của biến Unit price nằm gần đường thẳng, cho thấy dữ liệu có thể phân phối theo phân phối chuẩn. Tuy nhiên, có một số điểm dữ liệu nằm dưới đường thẳng ở phía bên trái của trục y, cho thấy dữ liệu có thể có xu hướng phân tán nhiều hơn ở phía bên trái của giá trị trung bình.



Hình 2.18 Histplot với Kde của biến Unit price.

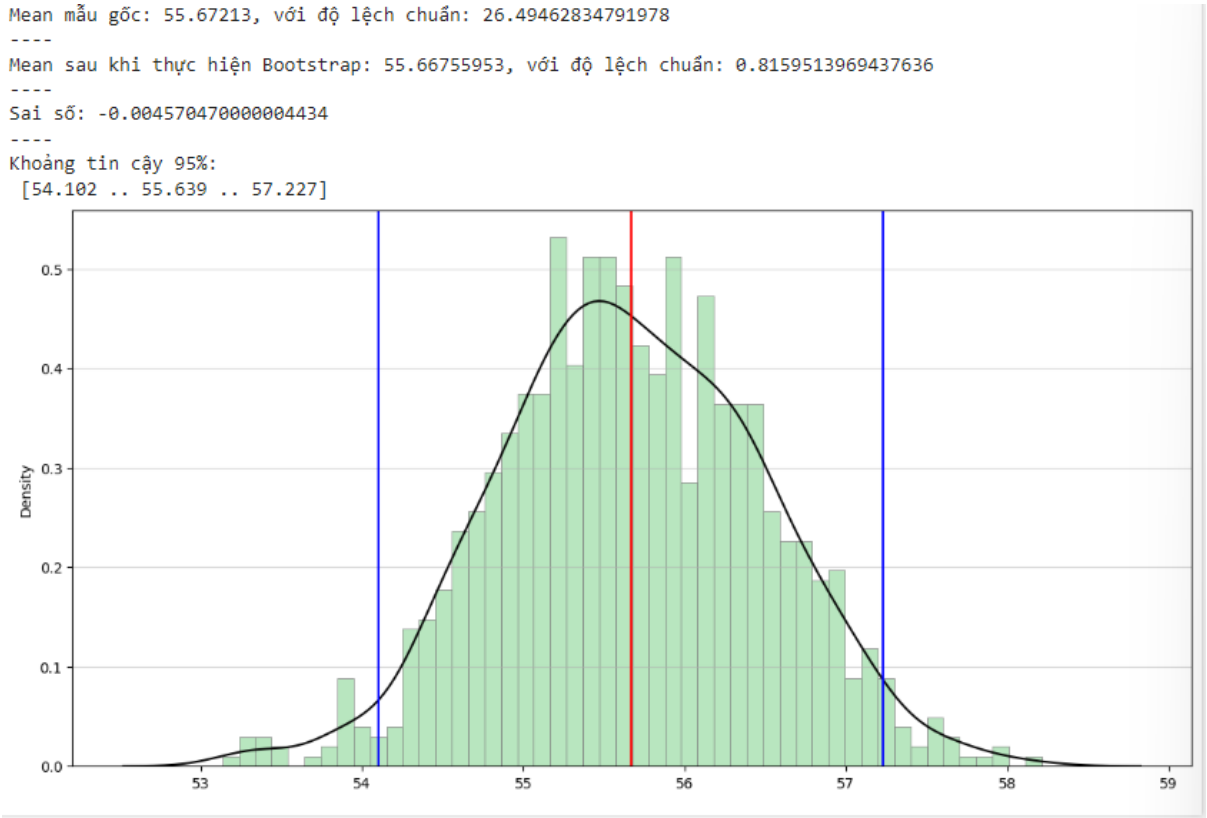
Nhận xét:

- Giá của các sản phẩm khá đa dạng trải dài từ 10 đến 100 USD.
- Giá của các sản phẩm có thể chia thành 3 phân khúc đó là: phân khúc giá thấp (10-33 USD); phân khúc giá trung bình (33-78 USD); phân khúc giá cao (từ 78 đến 100 USD).

Input:

```
#Áp dụng thuật toán bootstrap.
bootstrap_fun(df['Unit price'], np.mean,1000, 0.05)
```

Output:



Hình 2.19 Biểu đồ sau khi áp dụng thuật toán bootstrap cho biến Unit price.

Nhận xét:

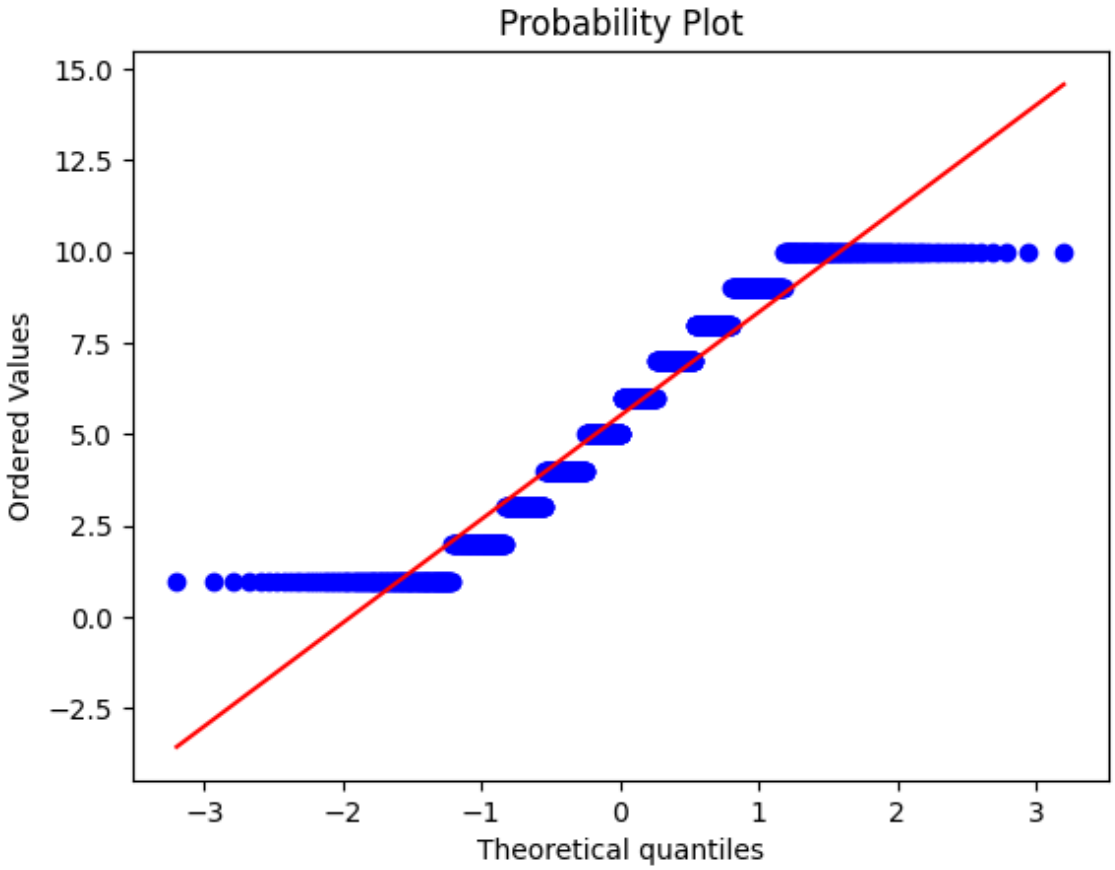
- Theo biểu đồ, giá trị trung bình mẫu gốc là 6.9727, với độ lệch chuẩn là 1.7185802943791215. Giá trị trung bình sau khi thực hiện Bootstrap là 6.9725664, với độ lệch chuẩn là 0.05522818203634806. Sai số giữa hai giá trị trung bình này là rất nhỏ, chỉ khoảng 0.00013359999999984495. Khoảng tin cậy 95% của giá trị trung bình là [6.865, 6.971, 7.083].

- Giá trị trung bình của các biến là khá ổn định, không có sự thay đổi đáng kể giữa giá trị trung bình mẫu gốc và giá trị trung bình sau khi thực hiện Bootstrap. Điều này cho thấy dữ liệu có tính chất phân phối chuẩn.

- Độ lệch chuẩn của các biến là khá cao, cho thấy dữ liệu có sự phân tán lớn xung quanh giá trị trung bình.

- Khoảng tin cậy 95% của giá trị trung bình khá hẹp, cho thấy giá trị trung bình có độ tin cậy cao.

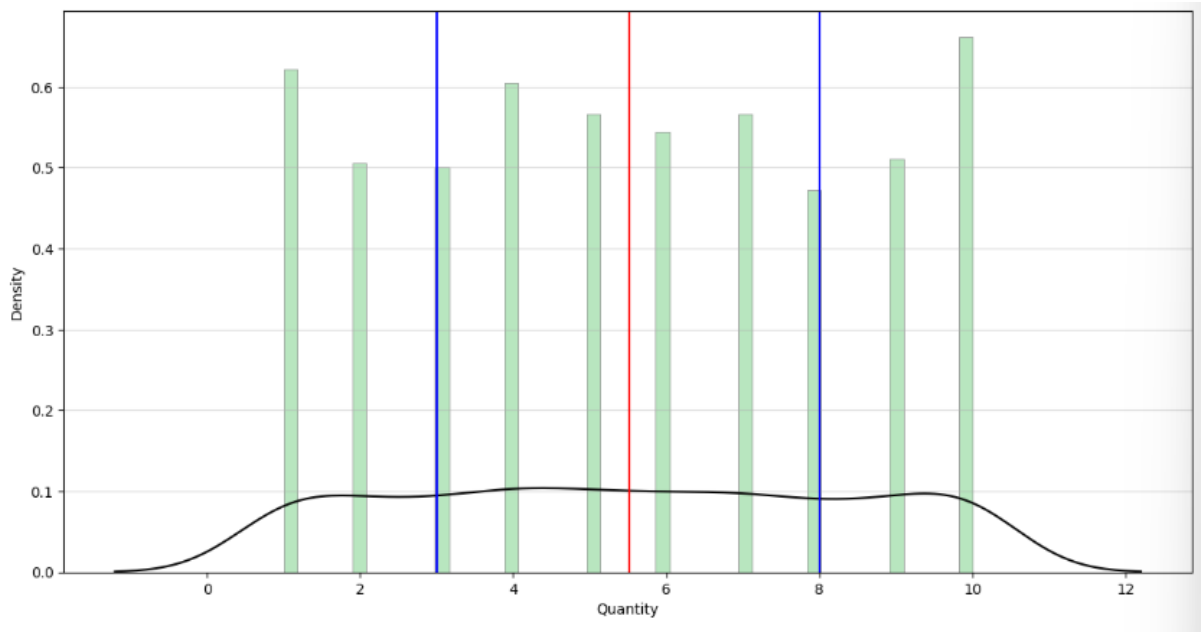
3.2. Biến Quantity:



Hình 2.20 Biểu đồ qq-plot cho biến Quantity.

Nhận xét:

- Trong biểu đồ này, các điểm dữ liệu của biến Quantity nằm gần đường thẳng, cho thấy dữ liệu có thể phân phối theo phân phối chuẩn.Tuy nhiên, có một số điểm dữ liệu nằm khá xa đường thẳng y = x. Các điểm dữ liệu này có thể là các giá trị ngoại lệ, bị ảnh hưởng bởi các biến số ngẫu nhiên khác.



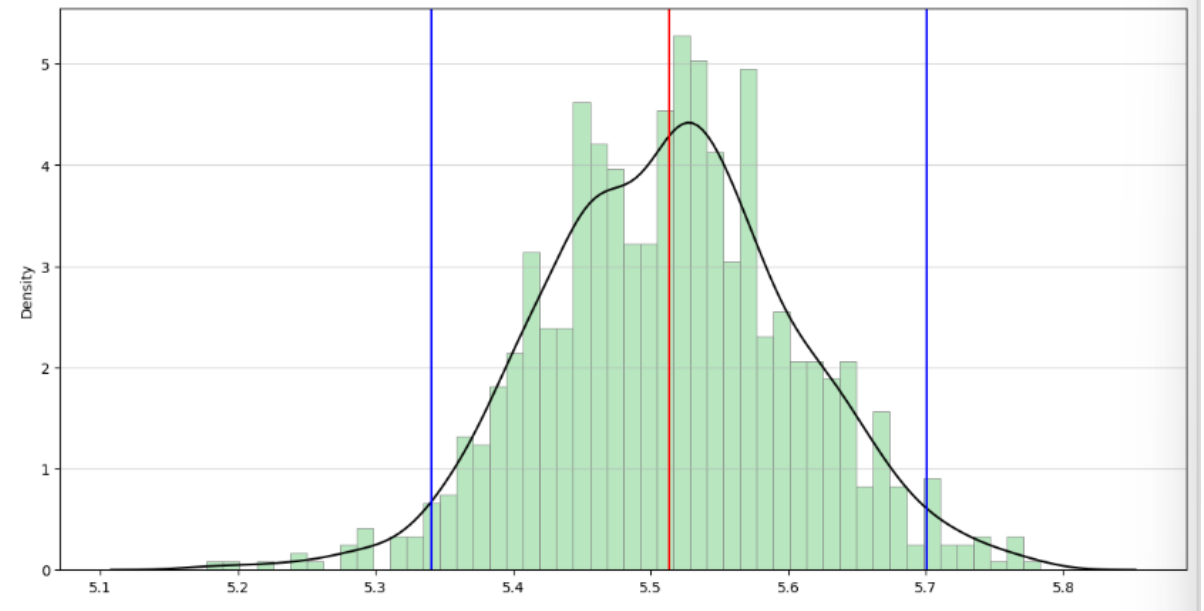
Hình 2.21 Histplot và kde cho biến Quantity.

Input:

```
#Áp dụng thực toán
bootstrap_fun(df['Quantity'], np.mean,1000, 0.05)
```

Output:

```
Mean mẫu gốc: 5.51, với độ lệch chuẩn: 2.923430595455696
----
Mean sau khi thực hiện Bootstrap: 5.513389999999999, với độ lệch chuẩn: 0.09201332457856308
----
Sai số: 0.00338999999999956
----
Khoảng tin cậy 95%:
 [5.341 .. 5.516 .. 5.701]
```



Hình 2.22 Biểu đồ sau khi áp dụng thuật toán bootstrap.

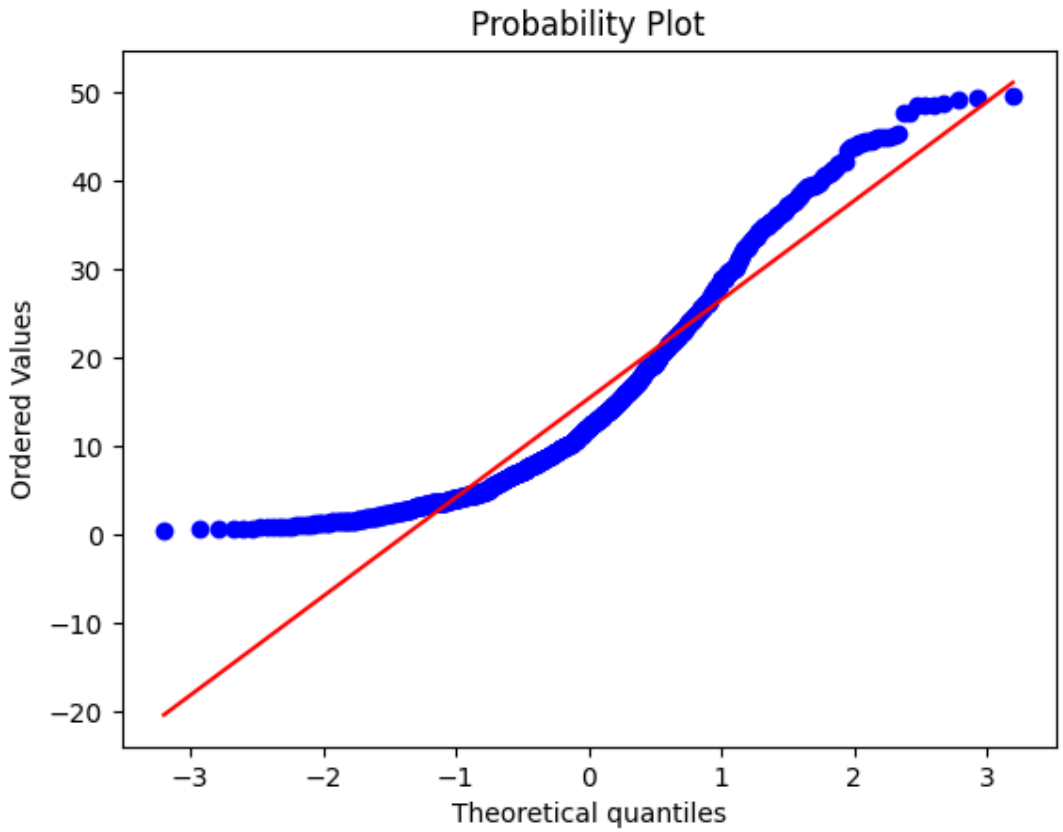
Nhận xét:

- Giá trị trung bình của biến quantity vẫn là khoảng 4.
- Giá trị phân tán của biến quantity đã giảm xuống khoảng 1.
- Các giá trị của biến quantity vẫn tập trung chủ yếu ở khoảng từ 2 đến 6.
- Số lượng các giá trị ngoại lệ đã giảm xuống.

Cụ thể hơn, biểu đồ cho thấy rằng có khoảng 50% các giá trị của biến quantity nằm trong khoảng từ 2,5 đến 5,5. Có khoảng 25% các giá trị nằm trong khoảng từ 1,5 đến 2,5 và khoảng 25% các giá trị nằm trong khoảng từ 5,5 đến 6,5.

Sự phân bố của các giá trị của biến quantity sau khi áp dụng thuật toán bootstrap có xu hướng tập trung hơn so với biểu đồ histplot ban đầu. Điều này cho thấy rằng thuật toán bootstrap đã giúp giảm thiểu sự ảnh hưởng của các giá trị ngoại lệ.

3.3. Biến Tax 5%:

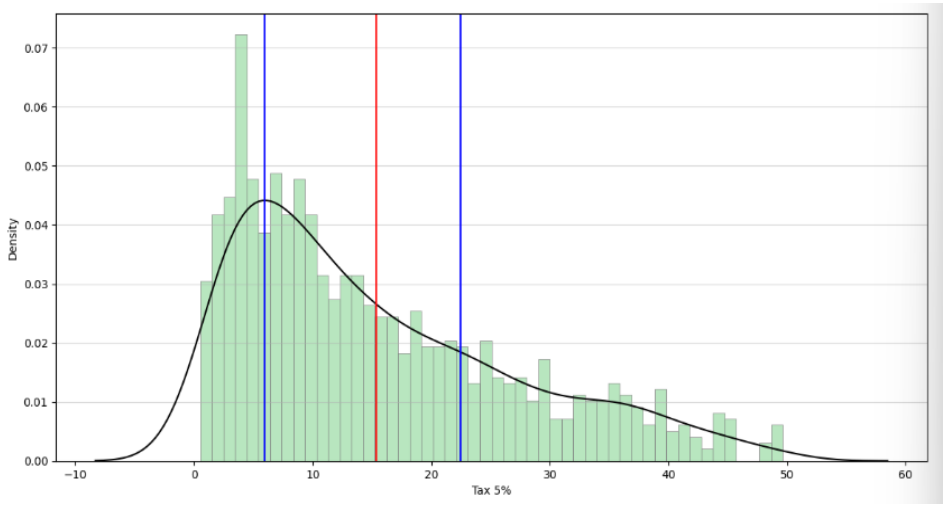


Hình 2.23: Biểu đồ qq-plot cho biến Tax 5%.

Nhận xét:

- Biểu đồ qq plot cho biến tax 5% cho thấy rằng:
    + Các điểm dữ liệu nằm khá gần đường thẳng lý tưởng.
    + Có một số điểm dữ liệu nằm ở phía dưới đường thẳng lý tưởng.

Sự phân bố của các điểm dữ liệu cho thấy rằng biến tax 5% có thể có phân phối gần với phân phối chuẩn. Tuy nhiên, cần lưu ý rằng có một số điểm dữ liệu nằm ở phía dưới đường thẳng lý tưởng. Điều này cho thấy rằng biến này có thể không hoàn toàn phân phối chuẩn.



Hình 2.24 Histplot và kde cho biến Tax 5%.

Nhận xét:

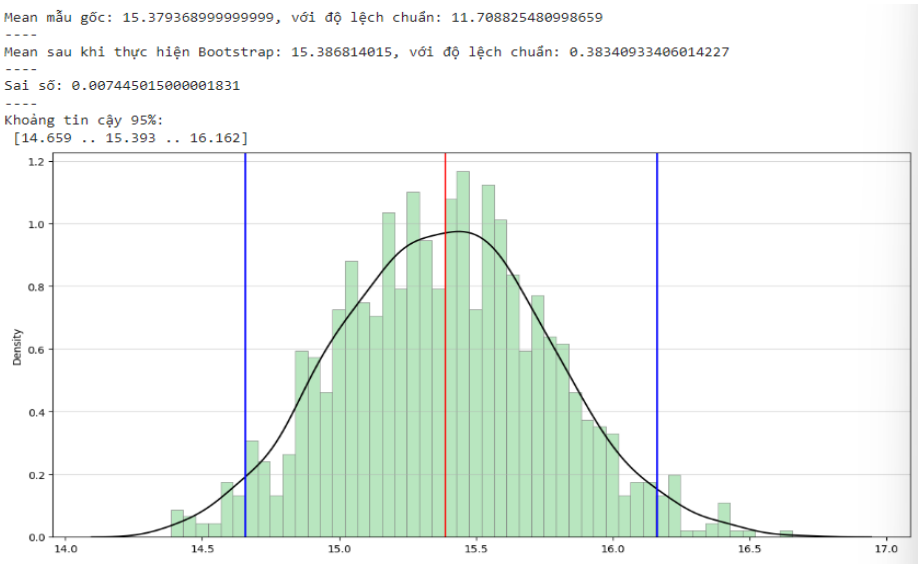
+ Giá trị của biến Tax 5% trải dài từ (0,50)
+ Giá trị trung bình khoảng 15.

Input:

```
#Áp dụng thuật toán bootstrap
bootstrap_fun(df['Tax 5%'], np.mean,1000, 0.05)
```

Output:

```
Mean mẫu gốc: 15.379368999999999, với độ lệch chuẩn: 11.708825480998659
----
Mean sau khi thực hiện Bootstrap: 15.386814015, với độ lệch chuẩn: 0.38340933406014227
----
Sai số: 0.007445015000001831
----
Khoảng tin cậy 95%:
 [14.659 .. 15.393 .. 16.162]
```

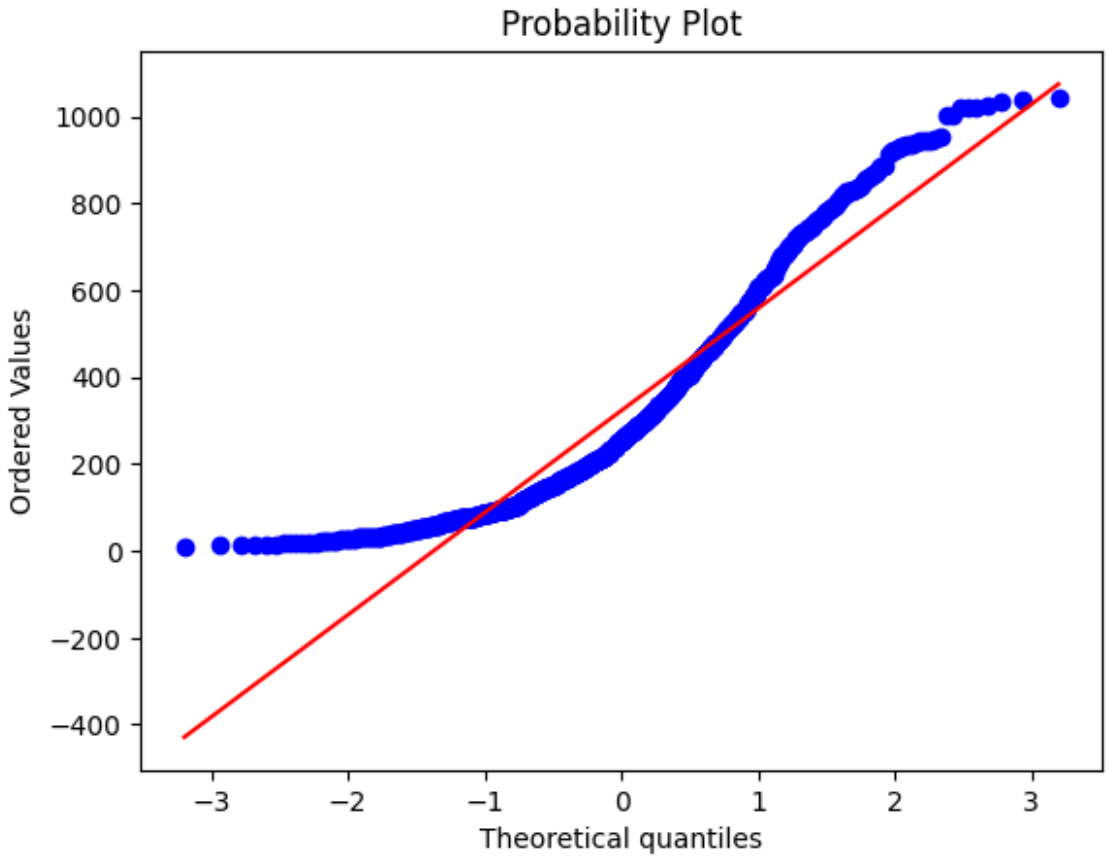


Hình2.25 Biểu đồ sau khi áp dụng thuật toán bootstrap.

Nhận xét:

- Biểu đồ cho thấy một phân phối chuẩn với giá trị trung bình là 15.386814015 và độ lệch chuẩn là 0.38340933406014227. Giá trị trung bình của phân phối sau khi áp dụng thuật toán Bootstrap không thay đổi so với giá trị trung bình của phân phối gốc. Độ lệch chuẩn của phân phối sau khi áp dụng thuật toán Bootstrap giảm đáng kể so với độ lệch chuẩn của phân phối gốc.

3.4. Biến Total:

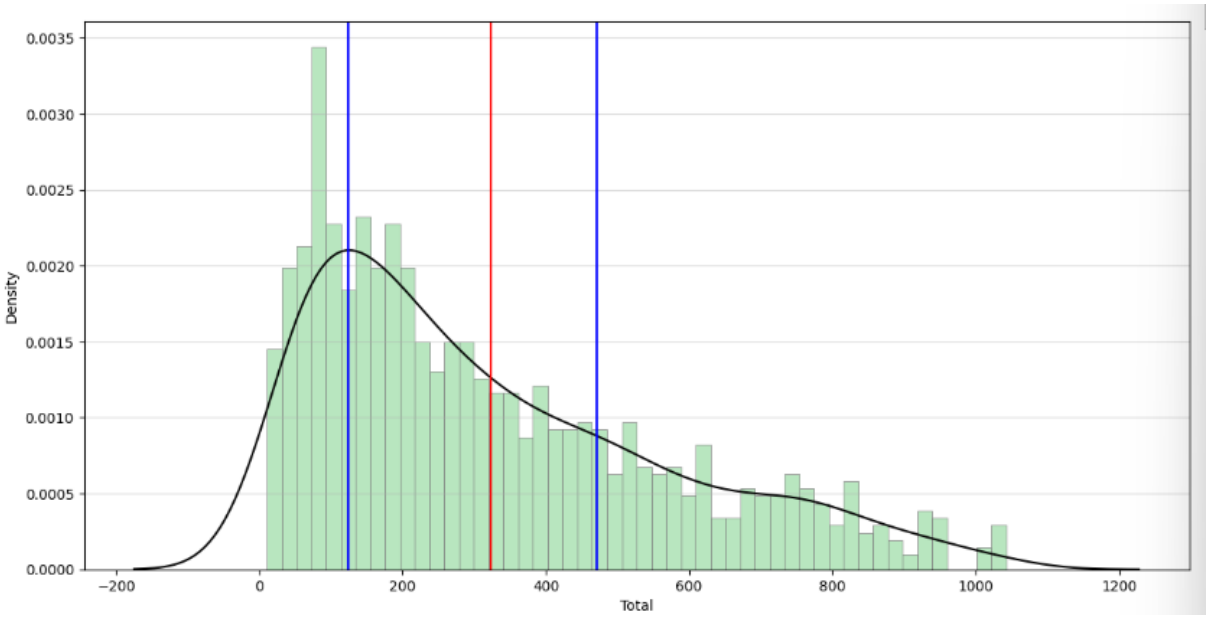


Hình 2.26 qq-plot cho biến Total.

Nhận xét:

- Một số giá trị quan sát nằm khá xa đường phân cách, đặc biệt là ở phần đuôi của phân phối. Điều này cho thấy rằng biến tổng chi tiêu có thể có một số điểm ngoại lai.

- Đường phân cách không hoàn toàn thẳng. Điều này cho thấy rằng biến tổng chi tiêu có thể không hoàn toàn phân phối chuẩn.



Hình 2.27 Histplot và kde cho biến Total.

Nhận xét:

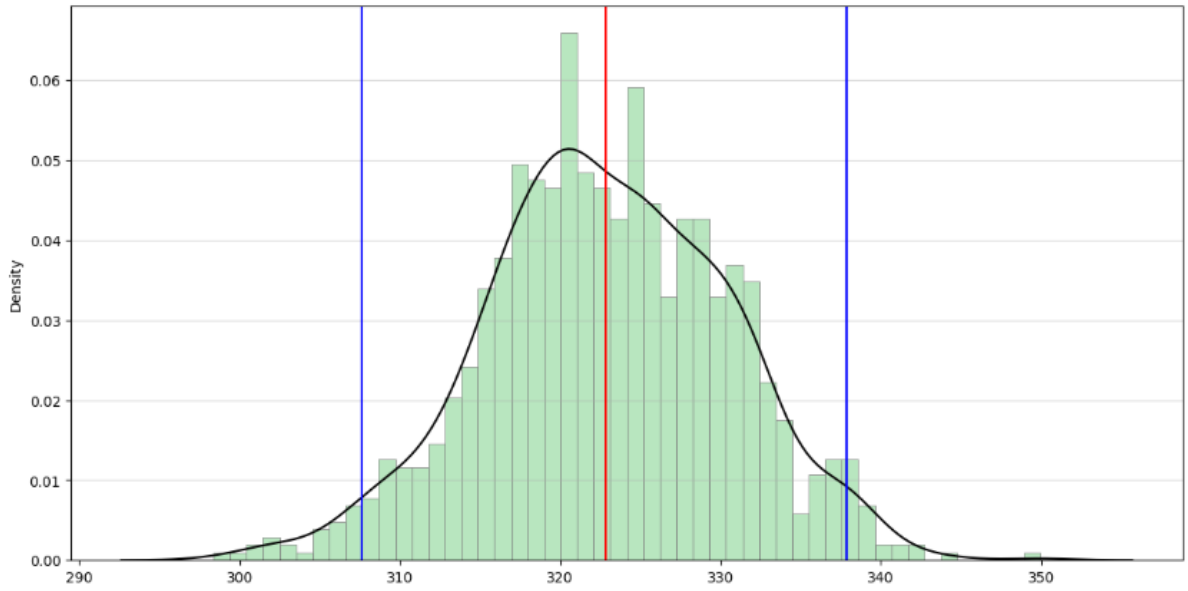
- Phân phối không đối xứng: Biểu đồ cho thấy rằng đuôi bên phải của phân phối dài hơn đuôi bên trái. Điều này có nghĩa là có nhiều giá trị tổng chi tiêu lớn hơn so với giá trị tổng chi tiêu nhỏ.
- Xu hướng tập trung vào các giá trị thấp: Phần lớn các giá trị tổng chi tiêu nằm trong khoảng từ 0 đến 400. Điều này cho thấy rằng biến tổng chi tiêu có xu hướng tập trung vào các giá trị thấp.
- Sự xuất hiện của các giá trị ngoại lai: Có một số giá trị tổng chi tiêu cao hơn đáng kể, lên tới hơn 1000. Các giá trị này có thể là do lỗi dữ liệu hoặc do sự tồn tại của các yếu tố khác không được xem xét trong mô hình.

Input:

```
#Áp dụng thuật toán bootstrap
bootstrap_fun(df['Total'], np.mean,1000, 0.05)
```

Output:

```
Mean mẫu gốc: 322.966749, với độ lệch chuẩn: 245.88533510097187
----
Mean sau khi thực hiện Bootstrap: 322.867514571, với độ lệch chuẩn: 7.601838420596366
----
Sai số: -0.09923442899997781
----
Khoảng tin cậy 95%:
 [307.650 .. 322.558 .. 337.837]
```



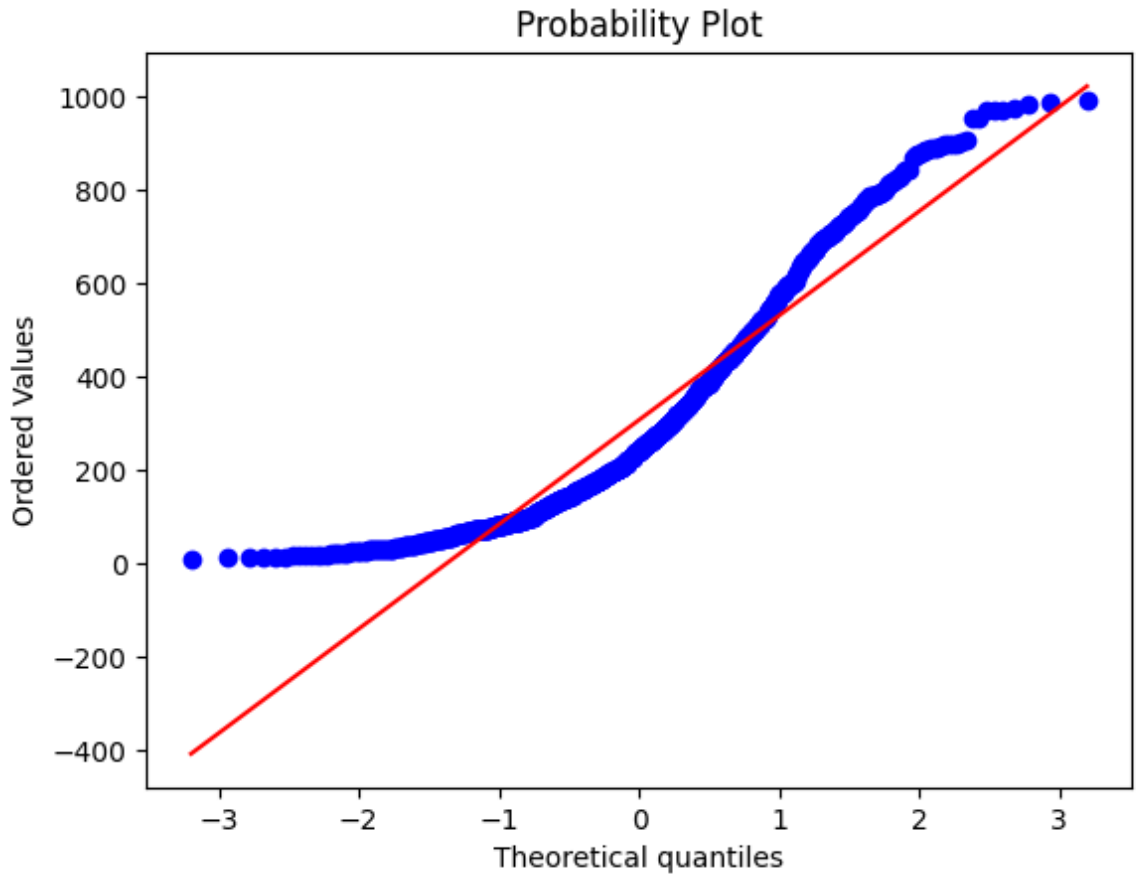
Hình 2.28 Biểu đồ sau khi áp dụng thuật toán bootstrap.

Nhận xét:

- Phân phối đối xứng hơn: Đuôi bên phải của phân phối đã ngắn hơn đáng kể, điều này cho thấy rằng có ít giá trị tổng chi tiêu lớn hơn so với trước đây.

- Phân phối tập trung hơn: Phần lớn các giá trị tổng chi tiêu nằm trong khoảng từ 0 đến 400, với ít giá trị tổng chi tiêu nằm ngoài khoảng này hơn so với trước đây.

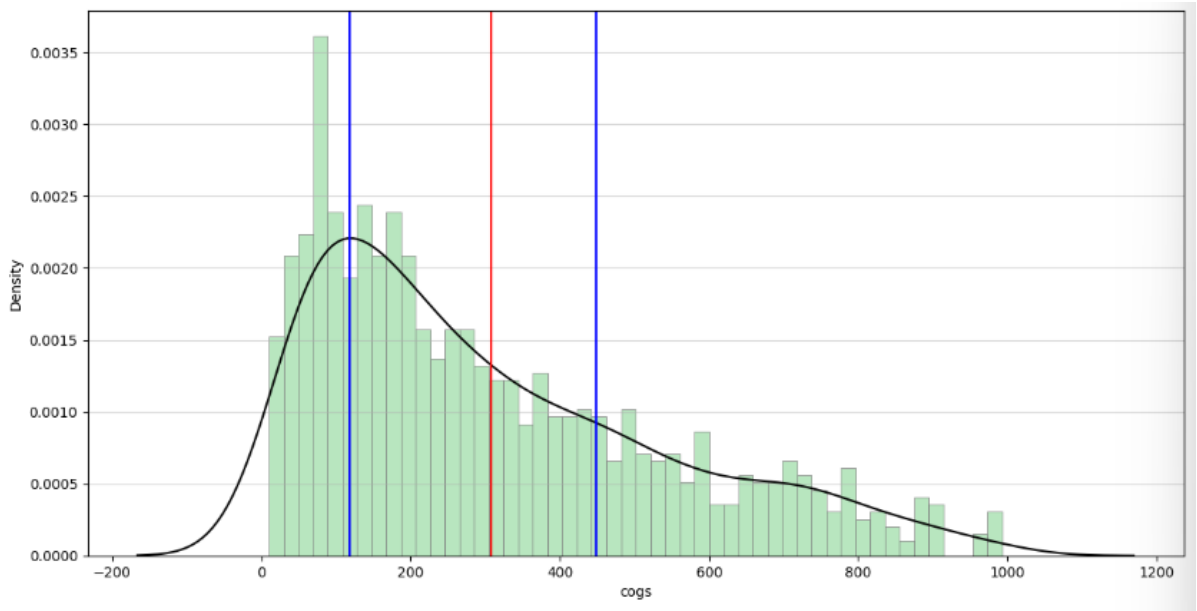
3.5. Biến cogs:



Hình 2.29 qq-plot cho biến cogs

Nhận xét:

- Một số giá trị quan sát nằm khá xa đường phân cách, đặc biệt là ở phần đuôi của phân phối. Điều này cho thấy rằng biến cogs có thể một số điểm ngoại lai.
- Đường phân cách không hoàn toàn thẳng. Điều này cho thấy rằng biến cogs có thể không hoàn toàn phân phối chuẩn.



Hình 2.30 Histplot và kde của biến cogs.

Nhận xét:

- Phân phối chuẩn: Biểu đồ cho thấy rằng phân phối của biến cogs có dạng hình chuông, với đỉnh nằm ở giá trị trung bình. Điều này cho thấy rằng biến cogs có thể được mô tả tốt bằng phân phối chuẩn.

- Xu hướng tập trung vào các giá trị thấp: Phần lớn các giá trị cogs nằm trong khoảng từ 0 đến 200. Điều này cho thấy rằng biến cogs có xu hướng tập trung vào các giá trị thấp.

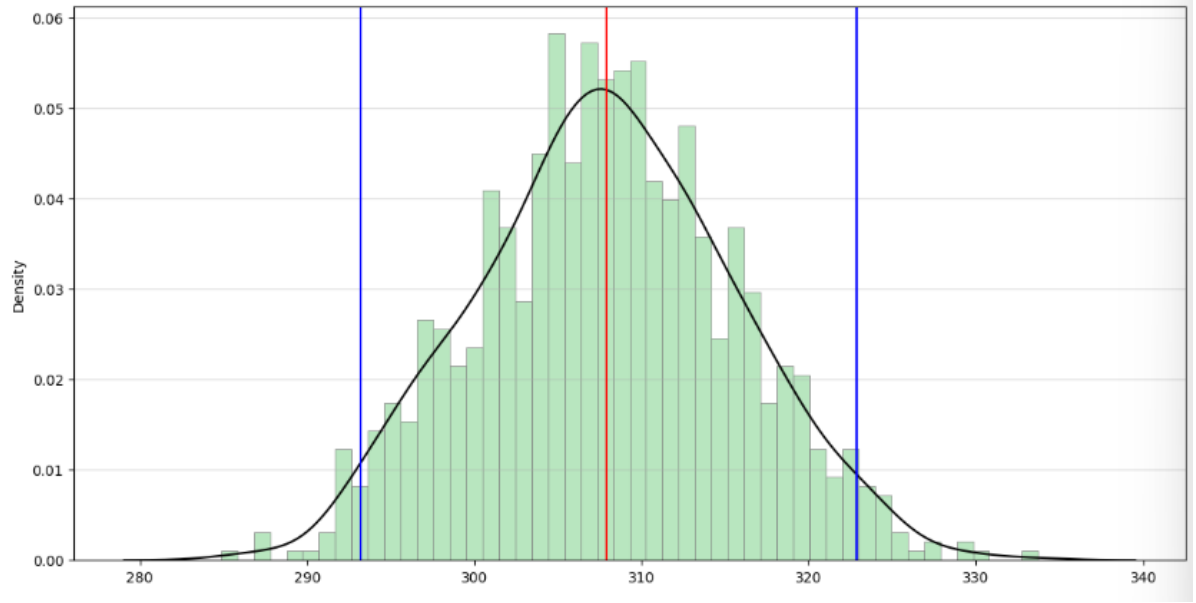
- Sự xuất hiện của các giá trị ngoại lai: Có một số giá trị cogs cao hơn đáng kể, lên tới hơn 1000. Các giá trị này có thể là do lỗi dữ liệu hoặc do sự tồn tại của các yếu tố khác không được xem xét trong mô hình.

Input:

```
#Áp dụng thuật toán bootstrap
bootstrap_fun(df['cogs'], np.mean,1000, 0.05)
```

Output:

```
Mean mẫu gốc: 307.58738, với độ lệch chuẩn: 234.1765096199732
----
Mean sau khi thực hiện Bootstrap: 307.88456626, với độ lệch chuẩn: 7.672256488501725
----
Sai số: 0.2971862599999895
----
Khoảng tin cậy 95%:
 [293.216 .. 307.822 .. 322.837]
```



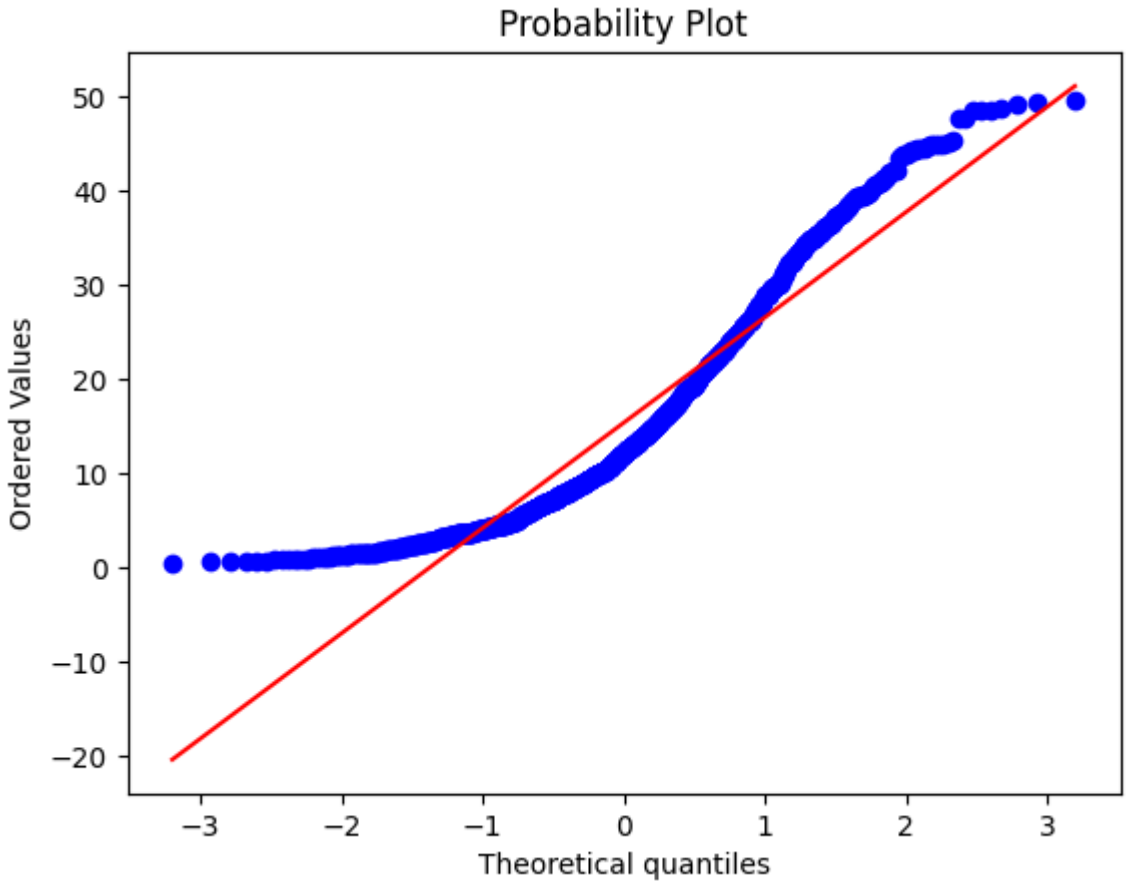
Hình 2.31 Biểu đồ sau khi áp dụng thuật toán bootstrap.

Nhận xét:

- Phân phối đối xứng hơn: Đuôi bên phải của phân phối đã ngắn hơn đáng kể, điều này cho thấy rằng có ít giá trị tổng chi tiêu lớn hơn so với trước đây.

- Phân phối tập trung hơn: Phần lớn các giá trị tổng chi tiêu nằm trong khoảng từ 0 đến 400, với ít giá trị tổng chi tiêu nằm ngoài khoảng này hơn so với trước đây.

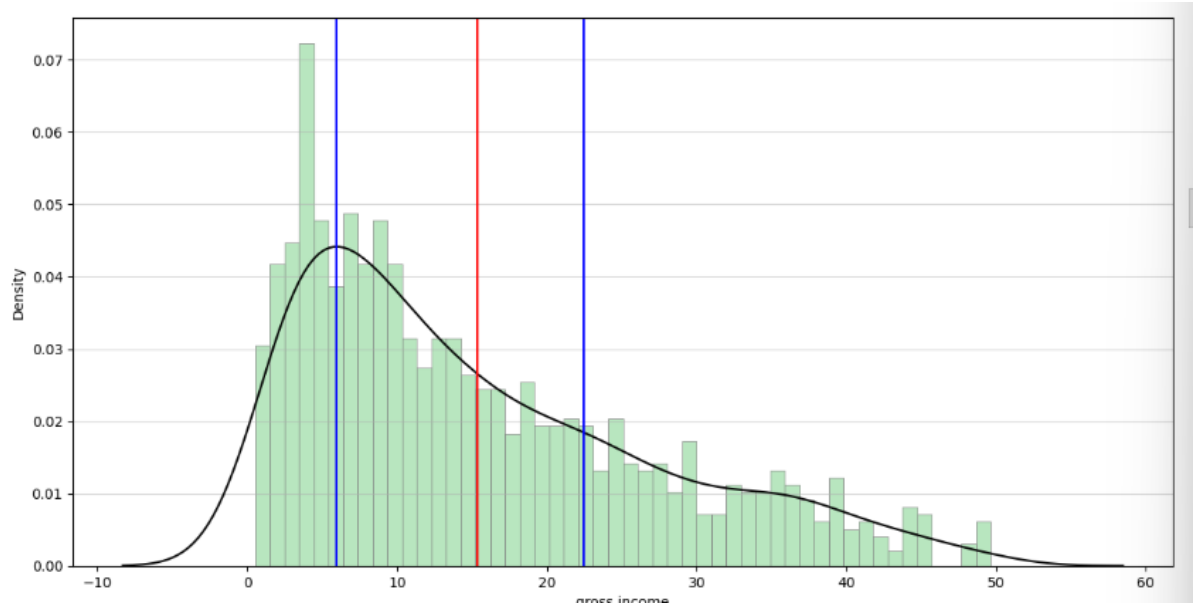
3.6. Biến gross income:



Hình 2.32 qq-plot cho biến gross income.

Nhận xét:

- Các giá trị quan sát nằm khá xa đường phân cách ở phần đuôi của phân phối. Điều này cho thấy rằng biến gross income có thể có một số điểm ngoại lai.
- Đường phân cách không hoàn toàn thẳng. Điều này cho thấy rằng biến gross income có thể không hoàn toàn phân phối chuẩn.



Hình 2.33 Histplot và kde cho biến gross income.

Nhận xét:

- Phân phối gần như chuẩn: Biểu đồ cho thấy rằng phân phối của biến gross income có dạng hình chuông, với đỉnh nằm ở giá trị trung bình. Điều này cho thấy rằng biến gross income có thể được mô tả tốt bằng phân phối chuẩn.

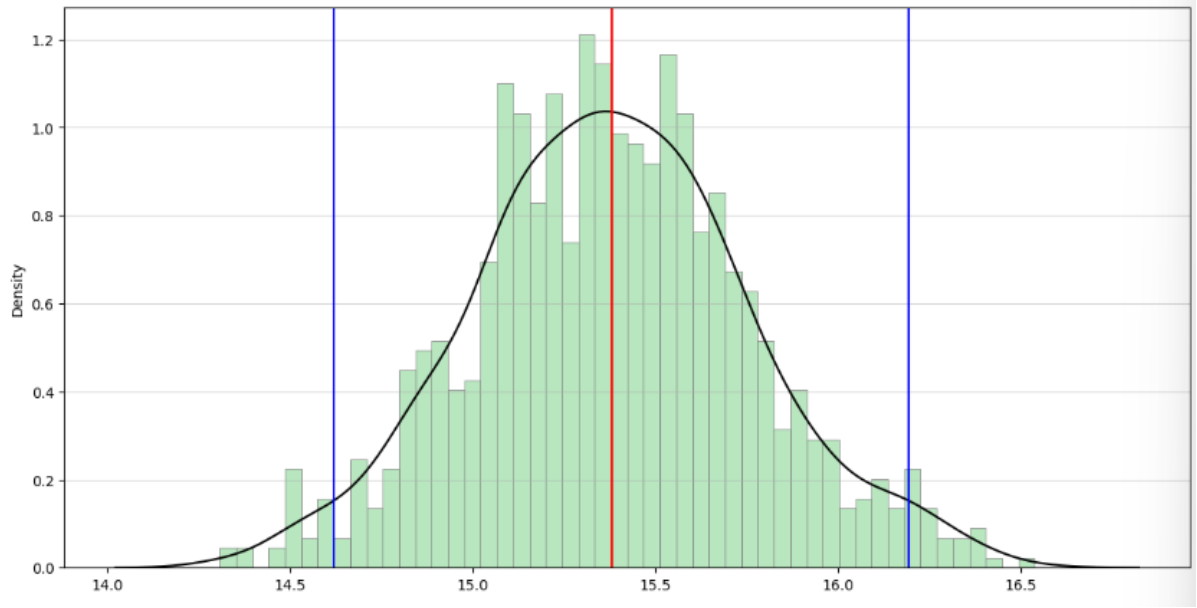
- Xu hướng tập trung vào các giá trị trung bình: Phần lớn các giá trị gross income nằm trong khoảng từ 0 đến 50. Điều này cho thấy rằng biến gross income có xu hướng tập trung vào các giá trị trung bình.

Input:

```
#Áp dụng thuật toán bootstrap
bootstrap_fun(df['gross income'], np.mean,1000, 0.05)
```

Output:

```
Mean mẫu gốc: 15.379368999999999, với độ lệch chuẩn: 11.708825480998659
----
Mean sau khi thực hiện Bootstrap: 15.380149746999999, với độ lệch chuẩn: 0.3788767682153691
----
Sai số: 0.0007807470000003036
----
Khoảng tin cậy 95%:
 [14.620 .. 15.374 .. 16.192]
```



Hình 2.34 Biểu đồ sau khi áp dụng bootstrap.

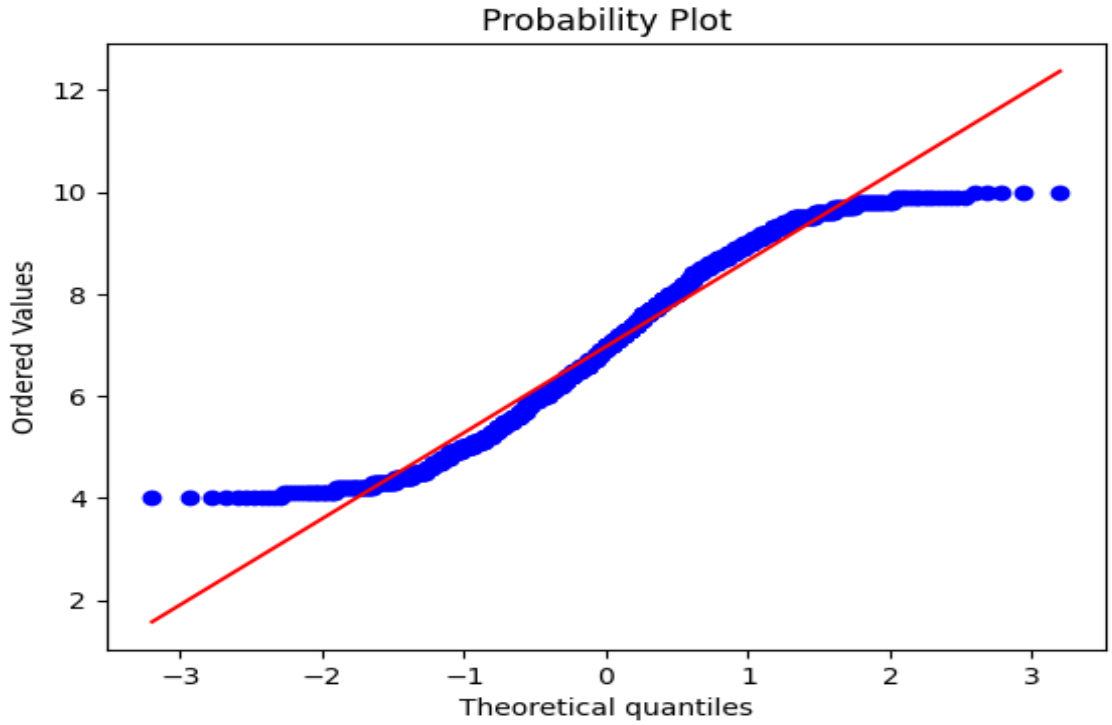
Nhận xét:

- Giá trị trung bình của dữ liệu không thay đổi. Giá trị trung bình của dữ liệu gốc là 15.379368999999999, và giá trị trung bình của dữ liệu sau khi áp dụng bootstrap là 15.380149746999999. Sự khác biệt giữa hai giá trị này là rất nhỏ, chỉ 0.0007807470000003036. Điều này cho thấy rằng thuật toán bootstrap không làm thay đổi giá trị trung bình của dữ liệu.

- Độ lệch chuẩn của dữ liệu giảm đáng kể. Độ lệch chuẩn của dữ liệu gốc là 11.708825480998659, và độ lệch chuẩn của dữ liệu sau khi áp dụng bootstrap là 0.3788767682153691. Sự giảm này là đáng kể, cho thấy rằng thuật toán bootstrap đã làm giảm sự biến thiên của dữ liệu.

- Khoảng tin cậy 95% của dữ liệu hẹp hơn. Khoảng tin cậy 95% của dữ liệu gốc là [14.0, 16.758737999999999], và khoảng tin cậy 95% của dữ liệu sau khi áp dụng bootstrap là [14.620, 16.192]. Sự hẹp lại của khoảng tin cậy cho thấy rằng thuật toán bootstrap đã làm tăng độ tin cậy của các ước tính về giá trị trung bình của dữ liệu.

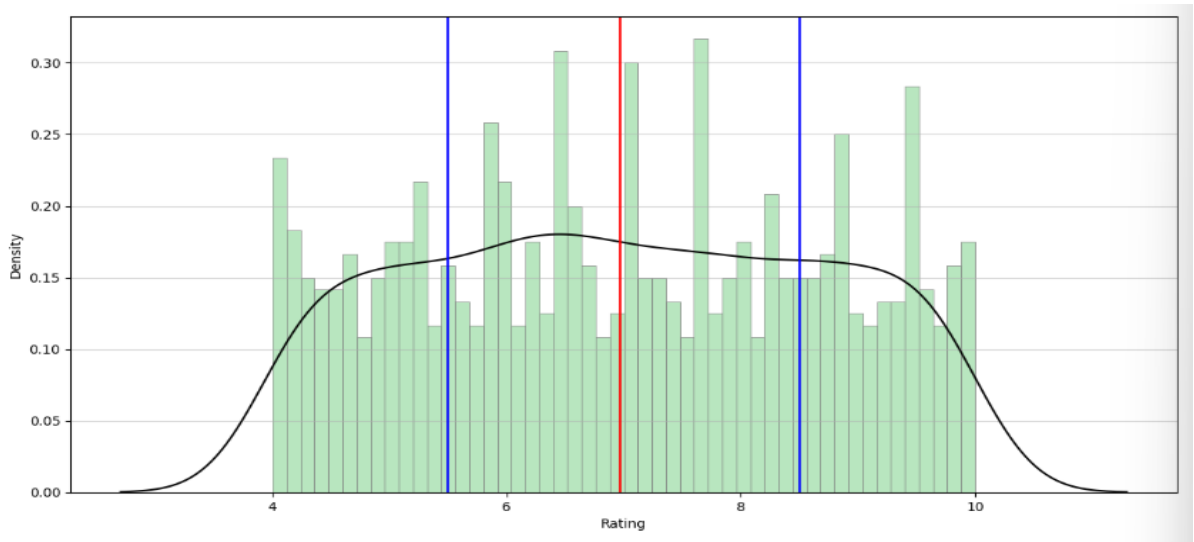
3.7. Biến Rating:



Hình 2.35 qq-plot cho biến Rating.

Nhận xét:

- Cụ thể, các điểm dữ liệu ở phần đuôi bên phải của biểu đồ nằm thấp hơn so với đường thẳng lý tưởng. Điều này cho thấy rằng có nhiều khách hàng đánh giá trải nghiệm của họ ở mức rất cao. Ngược lại, các điểm dữ liệu ở phần đuôi bên trái của biểu đồ nằm cao hơn so với đường thẳng lý tưởng. Điều này cho thấy rằng có ít khách hàng đánh giá trải nghiệm của họ ở mức rất thấp.



Hình 2.36 histplot và kde cho biến Rating.

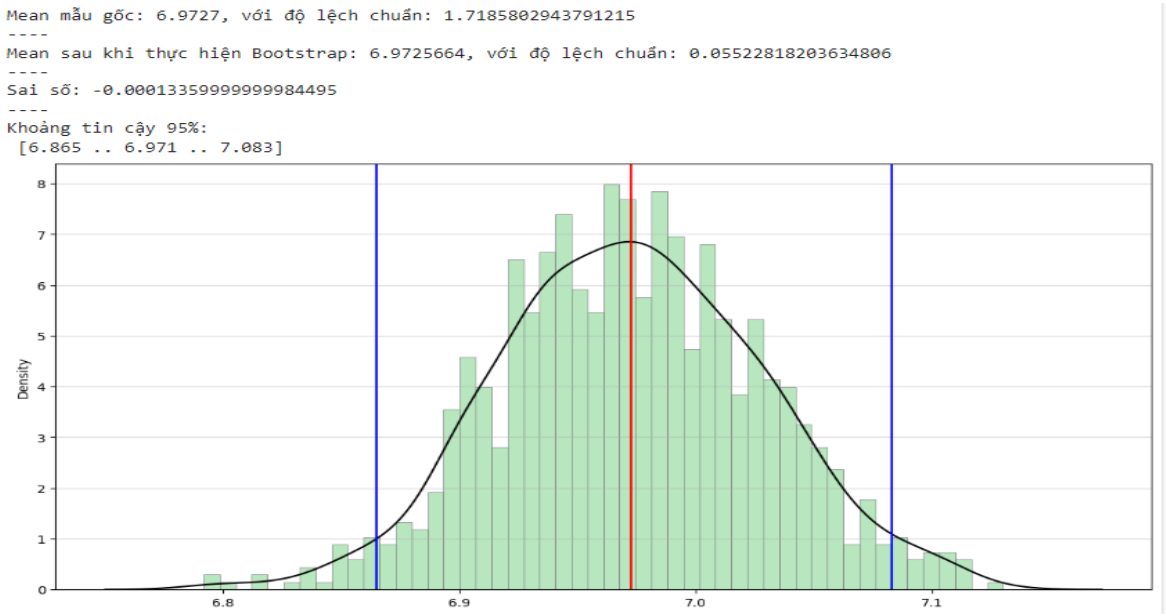
Nhận xét:

- Biểu đồ Histplot cho biến Rating cho thấy rằng biến này có phân phối không đối xứng lệch phải. Cụ thể, phần đuôi bên phải của biểu đồ cao hơn phần đuôi bên trái. Điều này có nghĩa là có nhiều khách hàng đánh giá trải nghiệm của họ ở mức rất cao, trong khi ít khách hàng đánh giá trải nghiệm họ ở mức rất thấp.

Input:

```
#Áp dụng thuật toán bootstrap
bootstrap_fun(df['Rating'], np.mean, 1000, 0.05)
```

Output:



Hình 2.37 Biểu đồ sau khi áp dụng bootstrap.

Nhận xét:

- Giá trị trung bình của dữ liệu không thay đổi. Giá trị trung bình của dữ liệu gốc là 6.9727, và giá trị trung bình của dữ liệu sau khi áp dụng bootstrap là 6.9725664. Sự khác biệt giữa hai giá trị này là rất nhỏ, chỉ 0.00013359999999984495. Điều này cho thấy rằng thuật toán bootstrap không làm thay đổi giá trị trung bình của dữ liệu.

- Độ lệch chuẩn của dữ liệu giảm đáng kể. Độ lệch chuẩn của dữ liệu gốc là 1.7185802943791215, và độ lệch chuẩn của dữ liệu sau khi áp dụng bootstrap là 0.05522818203634806. Sự giảm này là đáng kể, cho thấy rằng thuật toán bootstrap đã làm giảm sự biến thiên của dữ liệu.

- Khoảng tin cậy 95% của dữ liệu hẹp hơn. Khoảng tin cậy 95% của dữ liệu gốc là [6.865, 7.083], và khoảng tin cậy 95% của dữ liệu sau khi áp dụng bootstrap là [6.865, 6.971]. Sự hẹp lại của khoảng tin cậy cho thấy rằng thuật toán bootstrap đã làm tăng độ tin cậy của các ước tính về giá trị trung bình của dữ liệu.

4. Tiền xử lý dữ liệu

4.1. Kiểm tra dữ liệu bị thiếu

Tiến hành kiểm tra dữ liệu bị thiếu bằng isnull().sum() để kiểm tra và đếm số lượng giá trị rỗng xuất hiện trong bộ dữ liệu sau đó trả về tổng số các giá trị rỗng có được.

Input[4.1]:

```
for col in df.columns:
    pct_missing = df[col].isnull().mean()
    print(f'{pct_missing :.1%} - {col}')
```

Output[4.1]:

```
0.0% - Invoice ID
0.0% - Branch
0.0% - City
0.0% - Customer type
0.0% - Gender
0.0% - Product line
0.0% - Unit price
0.0% - Quantity
0.0% - Tax 5%
0.0% - Total
0.0% - Date
0.0% - Time
0.0% - Payment
0.0% - cogs
0.0% - gross margin percentage
0.0% - gross income
0.0% - Rating
```

Nhận xét:

- Theo kết quả thu thập được, ta thấy bộ dữ liệu được thu thập và ghi nhận một cách đầy đủ, không có giá trị bị thiếu nào xuất hiện.

4.2. Kiểm tra dữ liệu bị nhiễu

Ta thực hiện kiểm tra dữ liệu nhiễu bằng biểu đồ hộp boxplot. Trong thống kê mô tả, biểu đồ hộp hay boxplot (còn được gọi là biểu đồ hộp râu và ria) là một loại biểu đồ thường được sử dụng trong phân tích dữ liệu giải thích. Biểu đồ boxplot cho ta thấy được một cách trực quan sự phân phối dữ liệu số và độ lệch thông qua hiển thị phần tư dữ liệu (hoặc phần trăm) và trung bình, bao gồm: điểm tối thiểu, phần tư thứ nhất Q1 (phần tư thấp hơn), trung vị, phần tư thứ ba Q3 (phần tư lớn hơn) và điểm tối đa. Ngoài ra, biểu đồ hộp cung cấp cho chúng ta một bản tóm tắt trực quan về dữ liệu cho phép các nhà nghiên cứu nhanh chóng xác định các giá trị trung bình, độ phân tán của dữ liệu và cả các dấu hiệu của độ lệch. Chính vì những lý do đó, ta sẽ sử dụng biểu đồ boxplot để thực hiện kiểm tra các điểm dữ liệu bị nhiễu (outlier) nằm bên ngoài râu của boxplot.

Thực hiện truyền vào các dữ liệu định tính để quan sát nhiễu:

Input:

```
numerical_features = df.drop(['Invoice ID','Branch','City','Customer
type','Gender','Product line','Payment','Rating','Date','Time'], axis = 1)
```

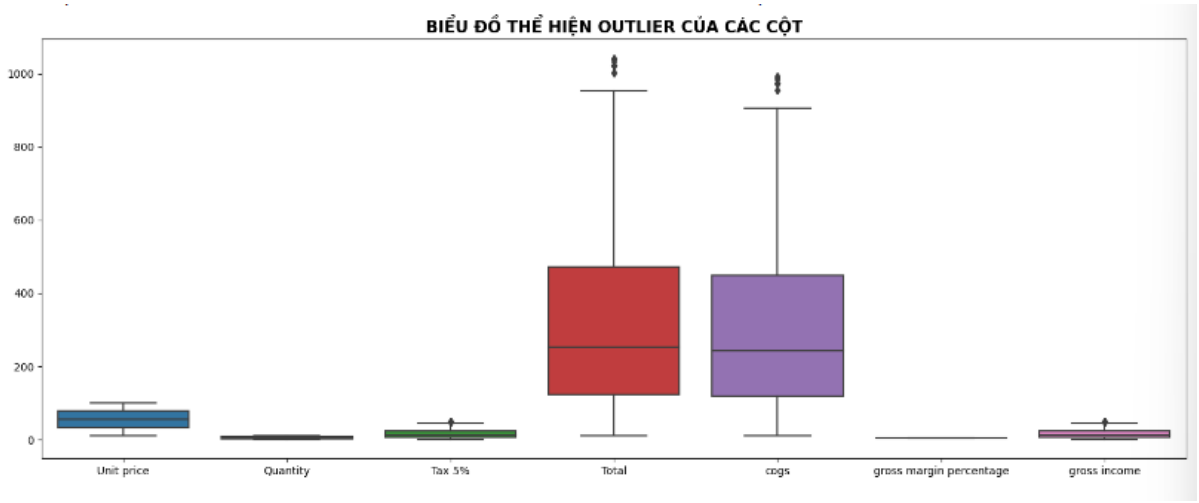
Output:

	Unit price	Quantity	Tax 5%	Total	cogs	gross margin percentage	gross income
0	74.69	7	26.1415	548.9715	522.83	4.761905	26.1415
1	15.28	5	3.8200	80.2200	76.40	4.761905	3.8200
2	46.33	7	16.2155	340.5255	324.31	4.761905	16.2155
3	58.22	8	23.2880	489.0480	465.76	4.761905	23.2880
4	86.31	7	30.2085	634.3785	604.17	4.761905	30.2085
...	...	...	...	...	...	...	...
995	40.35	1	2.0175	42.3675	40.35	4.761905	2.0175
996	97.38	10	48.6900	1022.4900	973.80	4.761905	48.6900
997	31.84	1	1.5920	33.4320	31.84	4.761905	1.5920
998	65.82	1	3.2910	69.1110	65.82	4.761905	3.2910
999	88.34	7	30.9190	649.2990	618.38	4.761905	30.9190

Input:

```
%matplotlib inline
plt.subplots(figsize = (19,7))
sns.boxplot(data = numerical_features)
plt.title('BIỂU ĐỒ THỂ HIỆN OUTLIER CỦA CÁC CỘT', fontsize = 15, fontweight = 'bold')
```

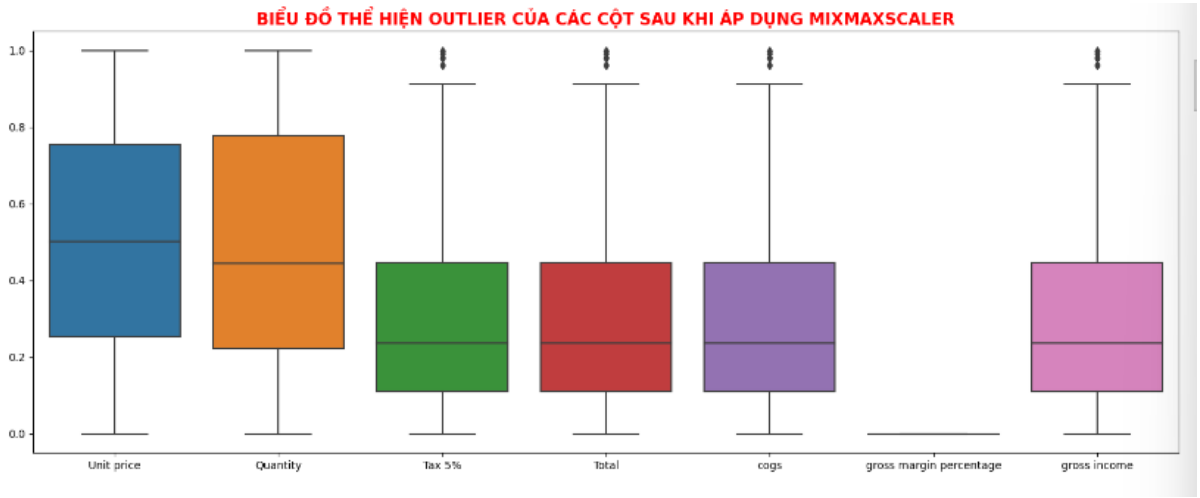
Biểu đồ:



Input:

```
# Chuẩn hóa dữ liệu bằng MinMaxScaler
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
sa = scaler.fit_transform(numerical_features)
sd = pd.DataFrame(sa, columns = numerical_features.columns)
plt. subplots(figsize = (19,7))
sns.boxplot(data = sd)
plt.title('BIỂU ĐỒ THỂ HIỆN OUTLIER CỦA CÁC CỘT SAU KHI ÁP DỤNG
MIXMAXSCALER', fontsize = 15, color = 'red', fontweight = 'bold')
```

Biểu đồ:



Biểu đồ thu thập được chưa được thể hiện một cách trực quan các dữ liệu bị nhiễu của bộ dữ liệu nên ta sẽ chuẩn hóa boxplot bằng cách sử dụng MinMaxScaler để có một biểu đồ trực quan hơn.

4.3. Kiểm tra giá trị trùng lặp

```
df.duplicated().sum()
```

```
0
```

4.4. Lọc giá trị nhiễu:

Để xử lý nhiễu, ta thực hiện thay thế nhiễu bằng các tham số median. Quan sát các biểu đồ hộp, ta có thể thấy được đa số các outlier đều nằm bên phải Q3 (bằng với phân vị của 75%) nên ta sẽ thực hiện tính toán chênh lệch tứ phân vị và tính giới hạn trên vừa phải (Reasonable Upper Bound).

```
columns = numerical_features.columns
q1_list = []
q3_list = []
median_list = []
for c in columns:
  c1 = df[c].quantile(.25)
  c2 = df[c].quantile(.5)
  c3 = df[c].quantile(.75)
  q1_list.append(c1)
  q3_list.append(c3)
  median_list.append(c2)
```

Áp dụng thông kê mô tả khai báo 3 ‘list’ tương ứng với Q1 ứng với 25%, Q3 với 75% và Q2 là 50%. Đặt x = Q3, y = Q1, áp dụng (Q3 + IQR * 1.5) với IQR = (Q3 – Q1).

Input:

```
upper_bound = [(x + (x - y) * 1.5) for x, y in zip(q3_list, q1_list)]
upper_bound
```

Output:

```
[145.525,
 15.5,
 47.2258125,
 991.7420625,
 944.5162500000001,
 4.761904762,
 47.2258125]
```

Sau khi đã đặt ngưỡng Q3 + IQR * 1.5, nếu giá trị của thuộc tính được kiểm tra lớn hơn giá trị giới hạn trên thi thay thế giá trị của thuộc tính đó bằng trung vị median.

```
i = 0
for c in columns:
  df[c] = np.where(df[c] > upper_bound[i], median_list[i], df[c])
  i += 1
```

Lúc này các dữ liệu bị nhiễu đã được thay thế bằng giá trị Median tương ứng

	Unit price	Quantity	Tax 5%	Total	cogs	gross margin percentage	gross income	Rating
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000
mean	55.672130	5.510000	15.049521	316.039941	300.990420	4.761905	15.049521	6.97270
std	26.494628	2.923431	11.271937	236.710686	225.438749	0.000000	11.271937	1.71858
min	10.080000	1.000000	0.508500	10.678500	10.170000	4.761905	0.508500	4.00000
25%	32.875000	3.000000	5.924875	124.422375	118.497500	4.761905	5.924875	5.50000
50%	55.230000	5.000000	12.084000	253.764000	241.680000	4.761905	12.084000	7.00000
75%	77.935000	8.000000	22.041000	462.861000	440.820000	4.761905	22.041000	8.50000
max	99.960000	10.000000	45.325000	951.825000	906.500000	4.761905	45.325000	10.00000

CHƯƠNG 3: PHÂN TÍCH VÀ BIỂU DIỄN CÁC BIẾN TRONG BỘ DỮ LIỆU

1. Phân tích các chi nhánh trong bộ dữ liệu

1.1. Siêu thị có bao nhiêu chi nhánh của hàng và mỗi chi nhánh có số lượng bao nhiêu?

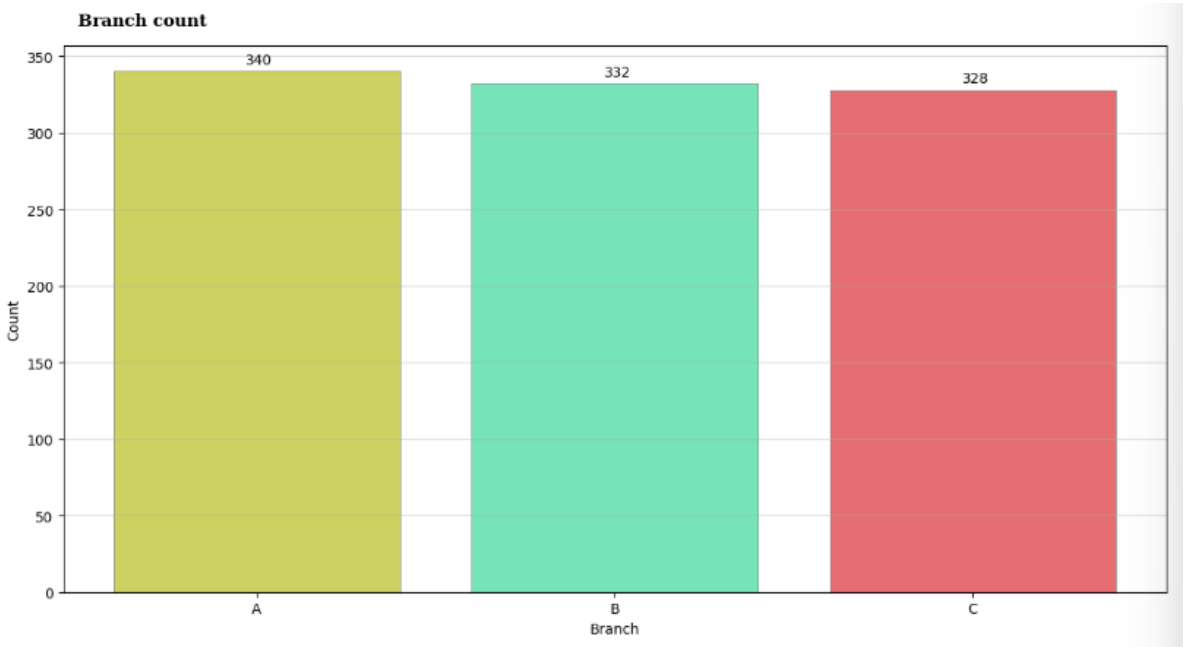
Input[1.1]:

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

Output[1.1]:

```
A    340
B    332
C    328
Name: Branch, dtype: int64
```

Biểu đồ:



Hình 3.1 Số lượng chi nhánh trong siêu thị

Nhận xét:

- Bộ dữ liệu với 1000 quan sát về siêu thị, ta có thể thấy được siêu thị có ở 3 chi nhánh A, B, C và số quan sát của các chi nhánh này trong bộ dữ liệu là ngang nhau. Trong đó lần lượt chi nhánh A chiếm 340, chi nhánh B chiếm 332 và thấp nhất là chi nhánh C chiếm 328 quan sát trong bộ dữ liệu.

1.2. Các chi nhánh tập trung ở những thành phố nào?

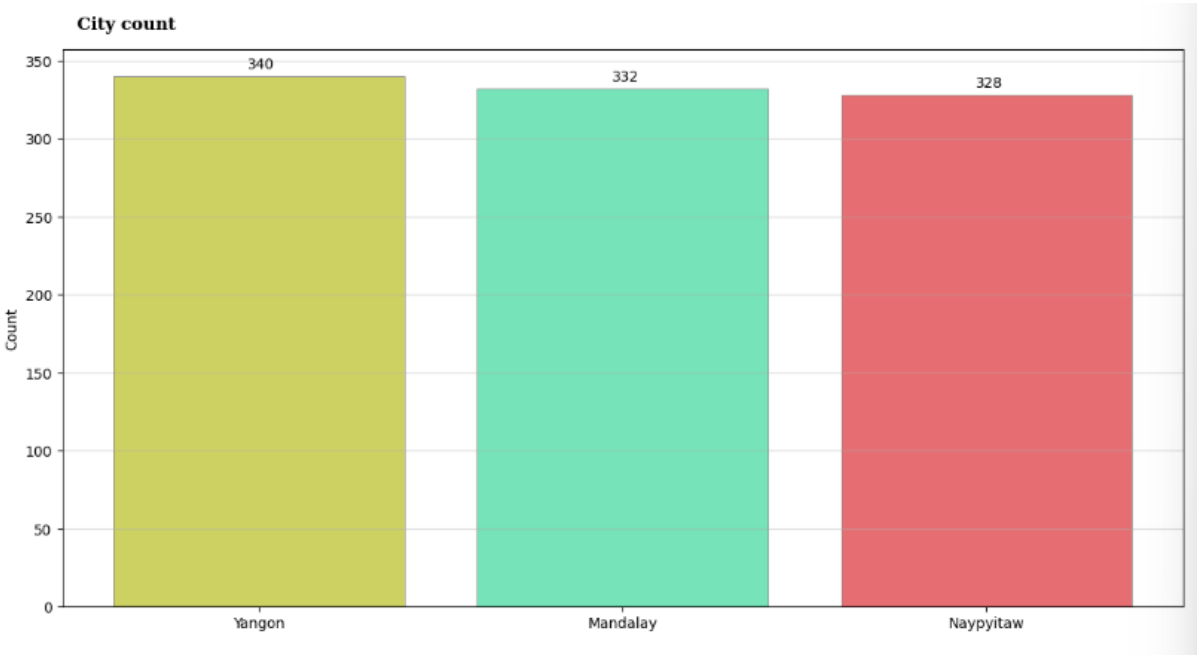
Input[1.2]:

```
city_counts = df["City"].value_counts()
city_counts
```

Output[1.2]:

```
Yangon       340
Mandalay     332
Naypyitaw    328
Name: City, dtype: int64
```

Biểu đồ:



Hình 3.2 Biểu đồ thể hiện số lượng thành phố

Nhận xét:

- Các chi nhánh tập trung nhiều nhất ở thành phố Yangon, tiếp đến là Mandalay và cuối cùng là Naypyitaw

1.3. Số lượng khách hàng mua hàng ở mỗi chi nhánh và thành phố là bao nhiêu?

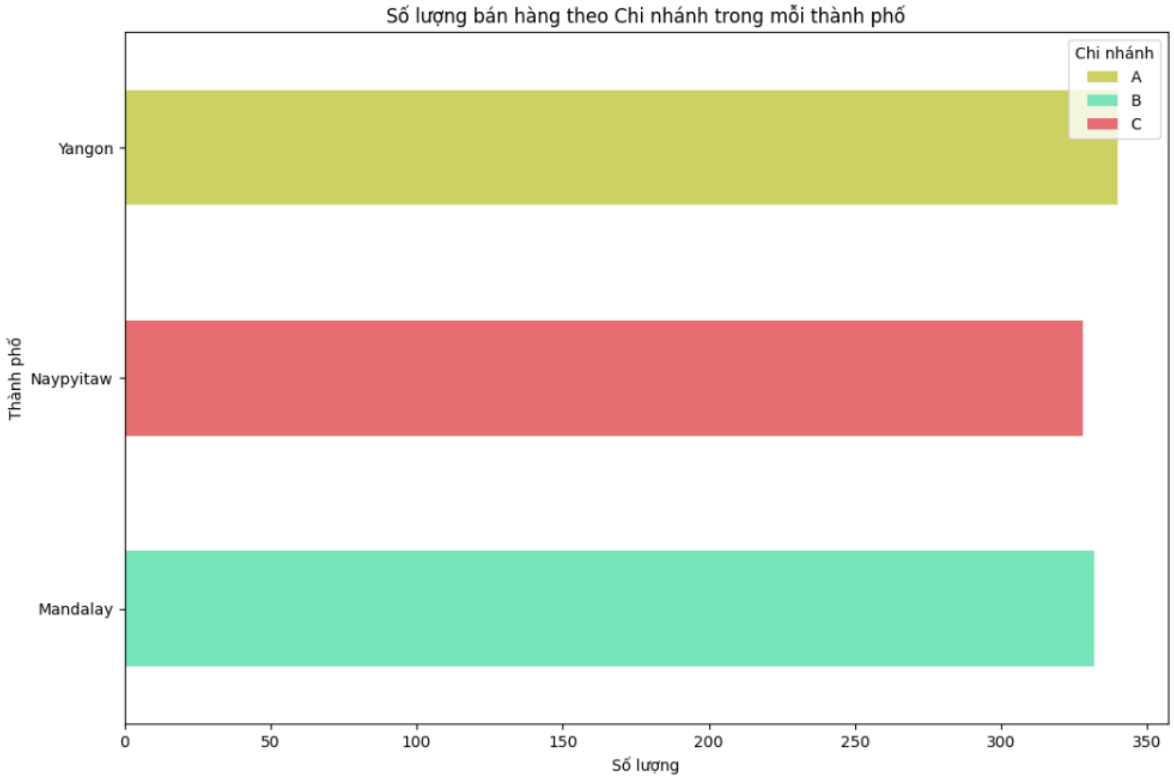
Input[1.3]:

```
city_branch_counts = df.groupby('City')['Branch'].value_counts()
city_branch_counts
```

Output[1.3]:

```
City       Branch
Mandalay   B         332
Naypyitaw  C         328
Yangon     A         340
Name: Branch, dtype: int64
```

Biểu đồ:



Hình 3.3 Biểu đồ thể hiện số lượng bán hàng theo chi nhánh ở mỗi thành phố

Nhận xét:

- Mỗi thành phố chứa một chi nhánh. Chi nhánh A nằm ở Yangon, chi nhánh B nằm ở Mandalay, chi nhánh C nằm ở Naypyitaw.

1.4. Thành phố nào có thu nhập gộp bán hàng cao nhất?

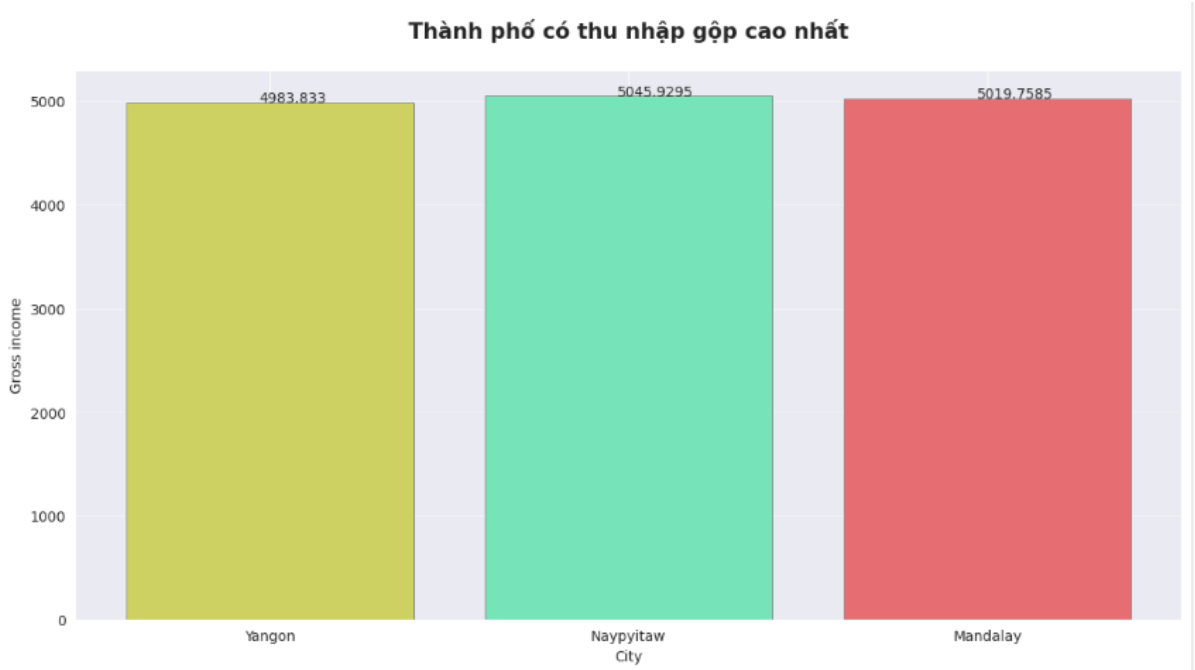
Input[1.4]:

```
city_gross_income = df.groupby('City')['gross income'].sum()

most_profitable_city = city_gross_income.idxmax()
total_income_most_profitable_city = city_gross_income.max()

print(f"Thành phố có thu nhập gộp cao nhất là {most_profitable_city} với tổng thu nhập gộp là {total_income_most_profitable_city}")
```

Output[1.4]:

Thành phố có thu nhập gộp cao nhất là Naypyitaw với tổng thu nhập gộp là 5045.9295

Biểu đồ:



Hình 3.4 Biểu đồ thể hiện thu nhập gộp mỗi thành phố

Nhận xét:

- Thành phố có thu nhập gộp cao nhất là Naypyitaw với tổng thu nhập gộp là 5045.9295

1.5. Tổng doanh thu bán hàng của mỗi chi nhánh và thành phố là bao nhiêu?

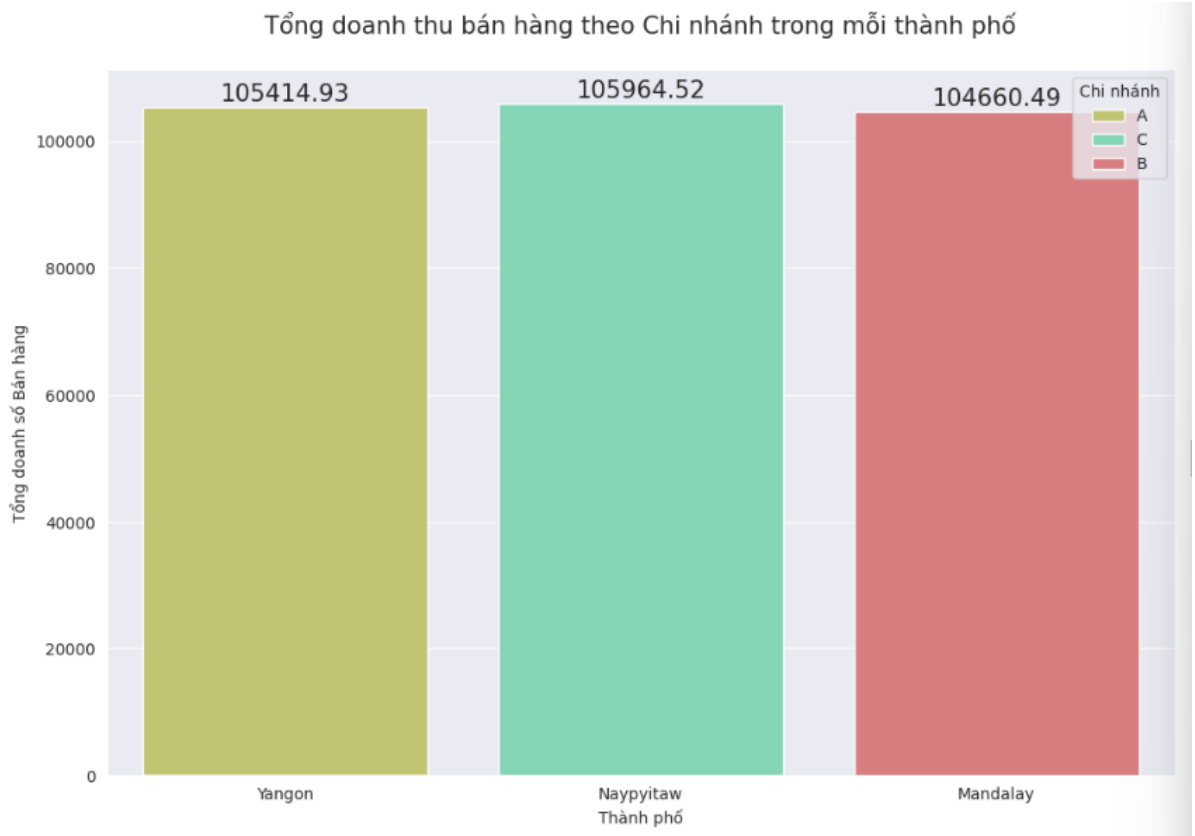
Input[1.5]:

```
city_branch_totals = df.groupby(['City', 'Branch'])['Total'].sum()
city_branch_totals
```

Output[1.5]:

```
City       Branch
Mandalay   B         104660.4930
Naypyitaw  C         105964.5195
Yangon     A         105414.9285
Name: Total, dtype: float64
```

Biểu đồ:



Hình 3.5 Biểu đồ tổng doanh thu bán hàng theo chi nhánh ở mỗi thành phố

Nhận xét:

- Tổng doanh thu của các chi nhánh ở các thành phố ta có thể thấy được thành phố Naypyitaw có doanh thu bán hàng cao nhất

2. Phân tích tệp khách hàng của siêu thị

2.1. Thông tin về khách hàng của siêu thị

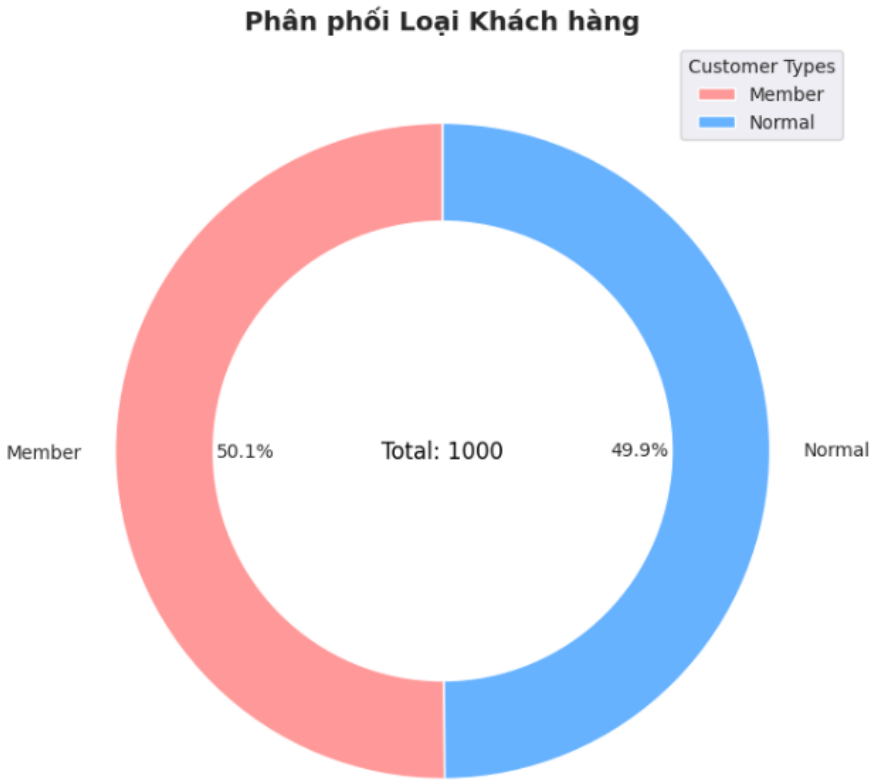
Input[2.1]:

```
cus_counts = df['Customer type'].value_counts()
cus_counts
```

Output[2.2]:

```
Member    501
Normal    499
Name: Customer type, dtype: int64
```

Biểu đồ:



Hình 3.6 Biểu đồ thể hiện tỷ lệ từng loại khách hàng

Nhận xét:

- Số lượng thành viên siêu thị và không phải là thành viên của siêu thị ngang nhau trong đó thành viên của siêu thị nhiều hơn 0.1%

2.2. Kiểm định loại khách hàng khác nhau thì tổng chi tiêu có thay đổi không?

Input[2.2]:

```
#Kiểm định liệu với loại khách hàng khác nhau thì tổng chi tiêu có
thay đổi
Member=data1[data1['Customer type']=='Member']['Total']
Normal=data1[data1['Customer type']=='Normal']['Total']
f, p = st.f_oneway(Member, Normal)
print('H0:Không có sự thay đổi giữa các loại khách hàng')
print('Ha:Có sự thay đổi giữa các loại khách hàng')
alpha = 0.05
if (p < alpha):
  print(f'Trị số p = {p:.4f} < {alpha:.4f} cho nên bác bỏ H0')

else:
  print(f'Trị số p = {p:.4f} >= {alpha:.4f} không nên bác bỏ H0')
```

Output[2.2]:

```
H0:Không có sự thay đổi giữa các loại khách hàng
Ha:Có sự thay đổi giữa các loại khách hàng
Trị số p = 0.3430 >= 0.0500 không nên bác bỏ H0
```

Nhận xét:

- Không có đủ cơ sở để vác bỏ H0

2.3. Tỷ lệ khách hàng nam/nữ là bao nhiêu?

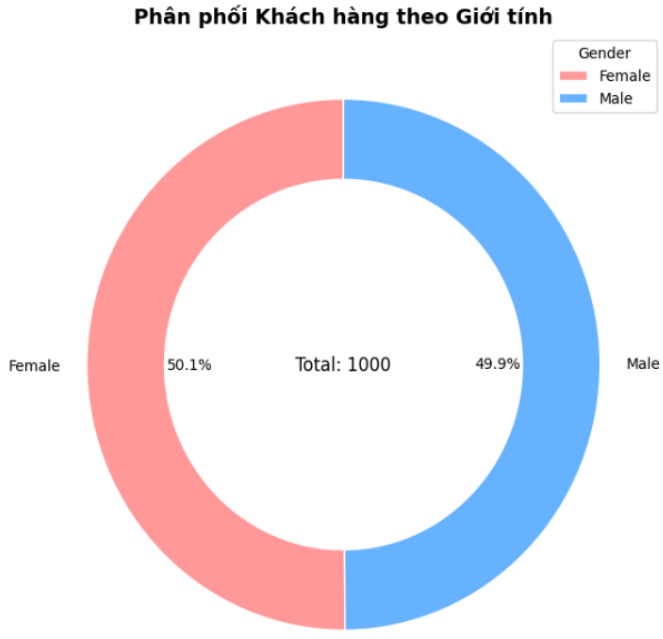
Input[2.3]:

```
label=df['Gender'].unique()
values=df['Gender'].value_counts().values
gender_counts = df['Gender'].value_counts()
gender_counts
```

Output[3]:

```
Female    501
Male      499
Name: Gender, dtype: int64
```

Biểu đồ:



Hình 3.7 Biểu đồ thể hiện phân phối khách hàng theo giới tính

Nhận xét:

- Số lượng khách hàng nam và số lượng khách hàng nữ là ngang nhau trong đó nữ giới nhiều hơn 0.1%

2.4. Kiểm định liệu với giới khác nhau thì tổng chi tiêu có thay đổi không?

Input[2.4]:

```
#Kiểm định liệu với giới khác nhau thì tổng chi tiêu có thay đổi
Male=data2[data2['Gender']=='Male']['Total']
Female=data2[data2['Gender']=='Female']['Total']
f, p = st.f_oneway(Male, Female)
print('H0:Không có sự thay đổi giữa các loại giới tính')
print('Ha:Có sự thay đổi giữa các loại giới tính')
alpha = 0.05
if (p < alpha):
  print(f'Trị số p = {p:.4f} < {alpha:.4f} cho nên bác bỏ H0')

else:
  print(f'Trị số p = {p:.4f} >= {alpha:.4f} không nên bác bỏ H0')
```

Output[2.4]:

```
H0:Không có sự thay đổi giữa các loại giới tính
Ha:Có sự thay đổi giữa các loại giới tính
Trị số p = 0.1883 >= 0.0500 không nên bác bỏ H0
```

Nhận xét:

- Không đủ điều kiện để bác bỏ H0

2.5. Giới tính của tệp khách hàng theo giới tính

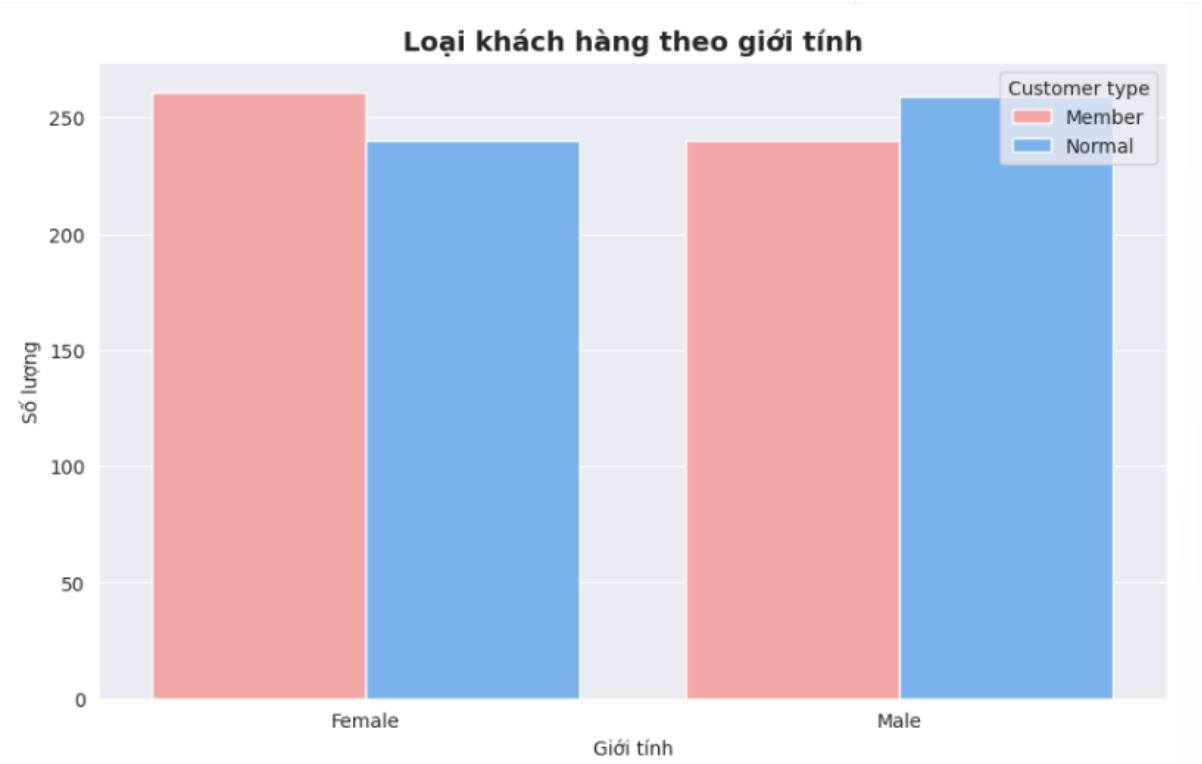
Input[2.5]:

```
gen_cus = df.groupby('Gender')["Customer type"].value_counts()
gen_cus
```

Output[2.5]:

```
Gender  Customer type
Female  Member           261
        Normal           240
Male    Normal           259
        Member           240
Name: Customer type, dtype: int64
```

Biểu đồ:



Hình 3.8 Biểu đồ thể hiện loại khách hàng theo giới tính

Nhận xét:

- Nam giới có số lượng khách hàng là thành viên nhiều hơn số lượng khách hàng không là thành viên.
- Nữ giới có số lượng khách hàng không là thành viên nhiều hơn số lượng khách hàng là thành viên.

3. Phân tích các dòng sản phẩm của siêu thị

3.1. Số lượng sản phẩm của từng dòng sản phẩm là bao nhiêu?

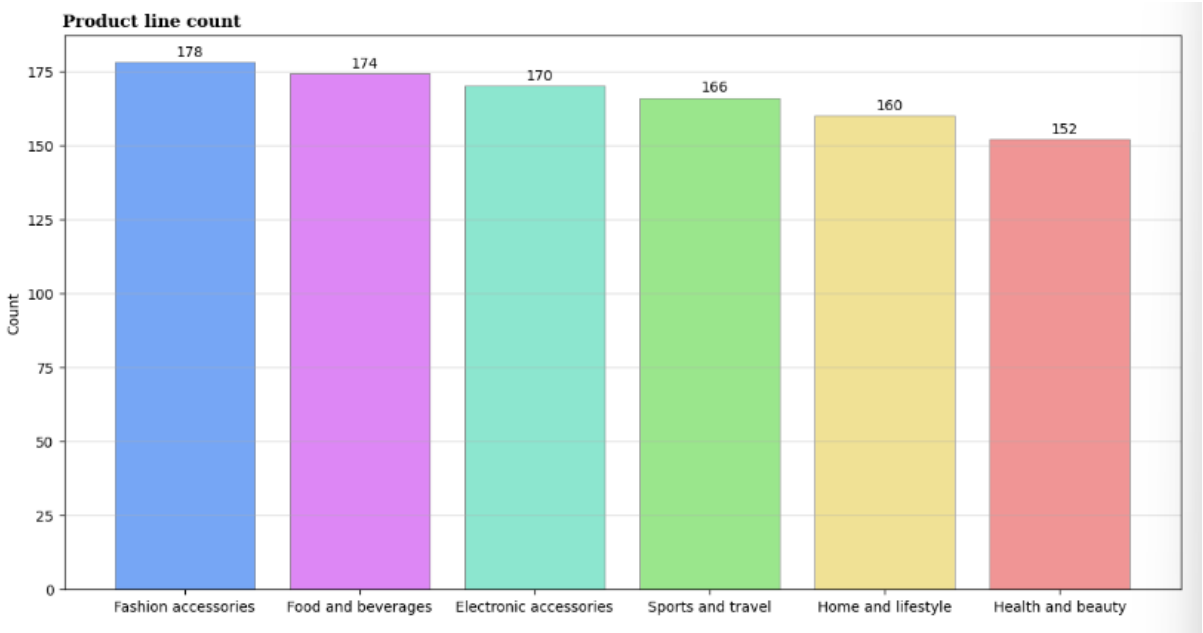
Input[3.1]:

```
product_count = df['Product
line'].value_counts().reset_index().set_index('index')
product_count
```

Output[3.1]:

index	Product line
Fashion accessories	178
Food and beverages	174
Electronic accessories	170
Sports and travel	166
Home and lifestyle	160
Health and beauty	152

Biểu đồ:



Hình 3.9 Số lượng dòng sản phẩm trong siêu thị

Nhận xét:

- Ta có thể tìm thấy số lượng dòng sản phẩm Phụ kiện thời trang nhiều nhất, kế đến là Thực phẩm và đồ uống và ít nhất là các dòng sản phẩm về Sức khỏe và sắc đẹp.

3.2. Tệp khách hàng mua các dòng sản phẩm

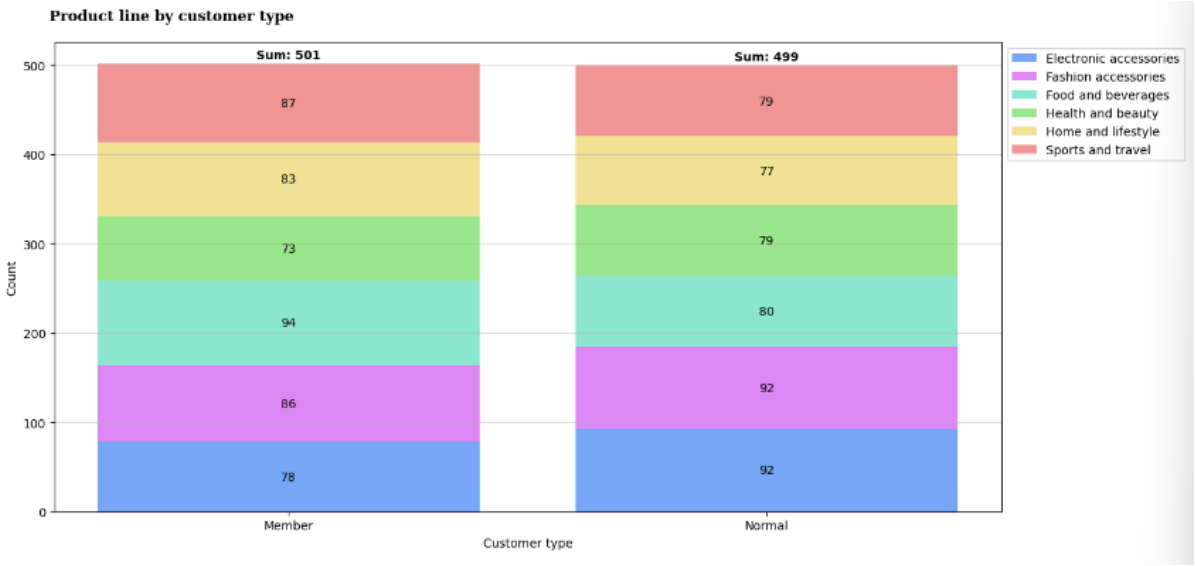
Input[3.2]:

```
product_customer = df[['Customer type', 'Product line']].groupby('Customer type')['Product line'].value_counts().unstack()
product_customer
```

Output[3.2]:

Product line	Electronic accessories	Fashion accessories	Food and beverages	Health and beauty	Home and lifestyle	Sports and travel
Customer type						
Member	78	86	94	73	83	87
Normal	92	92	80	79	77	79

Biểu đồ:



Hình 3.10 Biểu đồ thể hiện loại khách hàng mua các dòng sản phẩm

Nhận xét:

- Nhìn chung, khách hàng thành viên có xu hướng mua nhiều sản phẩm hơn khách hàng thông thường. Điều này có thể do khách hàng thành viên nhận được giảm giá hoặc ưu đãi khác
- Biểu đồ cũng cho thấy rằng thực phẩm và đồ uống là dòng sản phẩm phổ biến nhất đối với cả hai loại khách hàng.

3.3. Sản phẩm ở mỗi chi nhánh như thế nào?

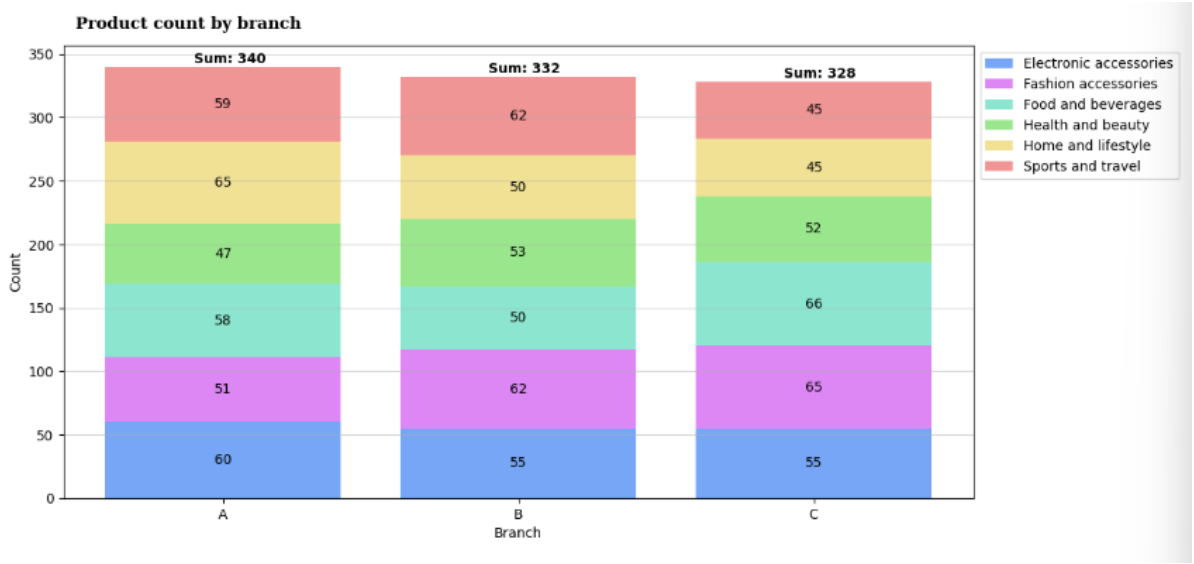
Input[3.3]:

```
product_branch = df[['Product line','Branch']].groupby('Branch')['Product line'].value_counts().unstack()
product_branch
```

Output[3.3]:

Product line	Electronic accessories	Fashion accessories	Food and beverages	Health and beauty	Home and lifestyle	Sports and travel
Branch						
A	60	51	58	47	65	59
B	55	62	50	53	50	62
C	55	65	66	52	45	45

Biểu đồ:



Nhận xét:

- Biểu đồ thể hiện số lượng sản phẩm đã bán cho từng chi nhánh và dòng sản phẩm.
- Nhìn chung, thấy rằng doanh số bán hàng của các chi nhánh khá tương đồng nhau. Tuy nhiên, có một số điểm khác biệt đáng chú ý: Chi nhánh A có doanh số bán hàng cao nhất và một số dòng sản phẩm có xu hướng bán chạy hơn ở chi nhánh này so với chi nhánh khác.

3.4. Giới tính của khách hàng mua các dòng sản phẩm

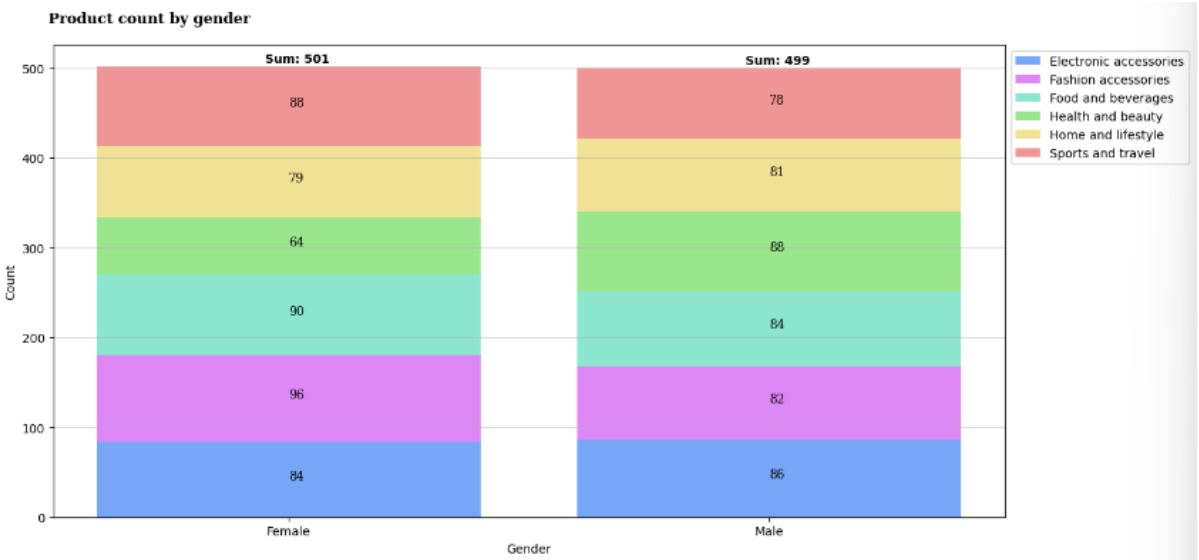
Input[3.4]:

```
product_gender = df[['Product line','Gender']].groupby('Gender')['Product line'].value_counts().unstack()
product_gender
```

Output[3.4]:

Product line	Electronic accessories	Fashion accessories	Food and beverages	Health and beauty	Home and lifestyle	Sports and travel
Gender						
Female	84	96	90	64	79	88
Male	86	82	84	88	81	78

Biểu đồ:



Nhận xét:

- Biểu đồ thể hiện số lượng sản phẩm đã bán cho từng giới tính và dòng sản phẩm.
- Nhìn chung, không có sự khác biệt đáng kể về sở thích sản phẩm giữa hai giới tính. Cả hai giới tính đều có xu hướng mua sắm các sản phẩm thực phẩm và đồ uống, phụ kiện thời trang và phụ kiện điện tử nhiều hơn so với các dòng sản phẩm khác.
- Tuy nhiên, có một số điểm khác biệt như khách hàng nữ có xu hướng mua sắm sản phẩm Phụ kiện thời trang nhiều hơn so với khách hàng nam, trong khi khách hàng nam có xu hướng mua sắm sản phẩm thể thao và du lịch nhiều hơn so với khách hàng nữ.

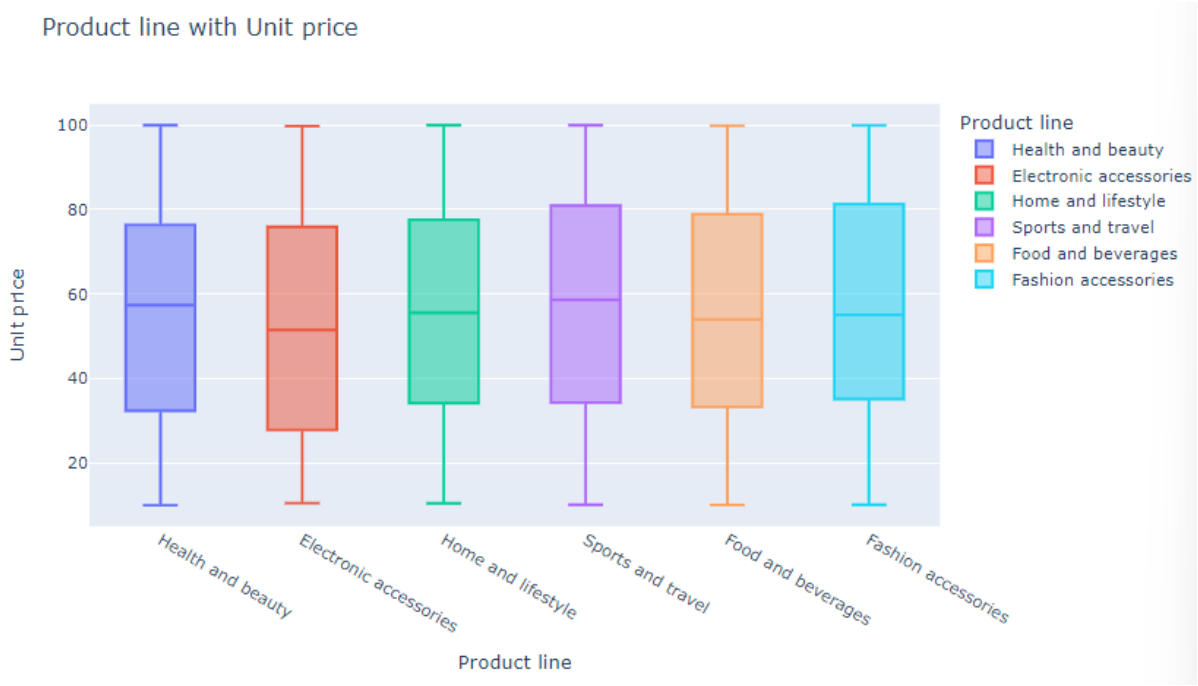
3.5. Giá của sản phẩm tính theo đơn vị đô la

Input[3.5]:

```
figure=px.box(df,x='Product line',color='Product line',y='Unit price',title='Product line with Unit price')
figure.show()
```

Output[3.5]:

Biểu đồ:



3.6. Tổng giá trung bình theo giới tính khách hàng khi mua mỗi dòng sản phẩm là bao nhiêu?

Input[3.6]:

```
df.pivot_table(index='Gender',columns='Product line',values='Total',aggfunc='mean')
```

Output[3.6]:

Product line	Electronic accessories	Fashion accessories	Food and beverages	Health and beauty	Home and lifestyle	Sports and travel
Gender						
Female	322.643125	300.851469	359.892283	290.015414	360.755677	316.209648
Male	316.691965	281.500646	273.499125	348.099460	275.375204	340.360327

Nhận xét: Ta có hành vi mua sắm của khách hàng theo giới tính:

- Nam giới có xu hướng chi tiêu nhiều hơn cho các sản phẩm điện tử và thể thao du lịch so với nữ giới. Điều này khá giống với ý nghĩ thông thường về sở thích của nam giới về các sản phẩm công nghệ và hoạt động thể thao.

- Nói chung, tổng số tiền chi tiêu của nam giới và nữ giới không chênh lệch đáng kể. Điều này cho thấy cả hai giới đều có vai trò quan trọng trong việc thúc đẩy doanh thu bán hàng của siêu thị.

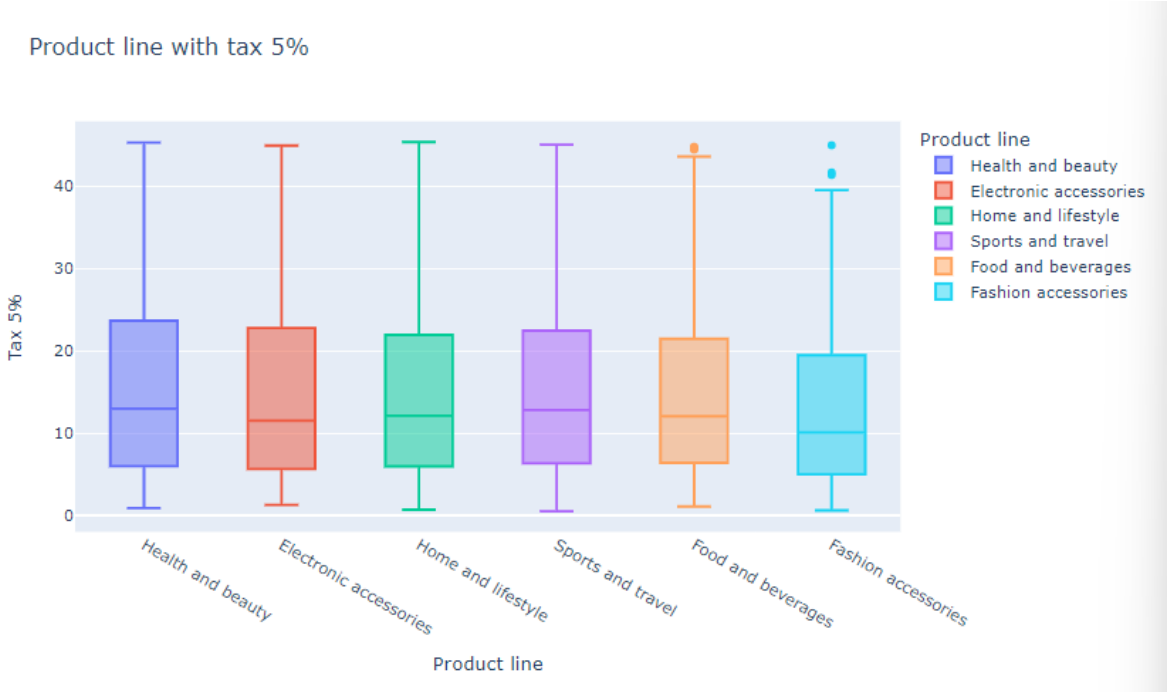
3.7. Người mua hàng phải chịu 5% thuế khi mua mỗi sản phẩm như thế nào?

Input[3.7]:

```
figure=px.box(df,x='Product line',color='Product line',y='Tax
5%',title='Product line with tax 5%')
figure.show()
```

Output[3.7]:

Biểu đồ:



Nhận xét:

- Thuế suất 5% được áp dụng cho tất cả các dòng sản phẩm. Tuy nhiên, có sự khác biệt về mức độ phân bố của thuế suất giữa các dòng sản phẩm.
- Dòng sản phẩm "Electronic accessories" có mức thuế suất 5% cao. Điều này có thể là do các sản phẩm điện tử thường có giá trị cao hơn so với các loại sản phẩm khác.

3.8. Doanh thu bán hàng theo từng dòng sản phẩm

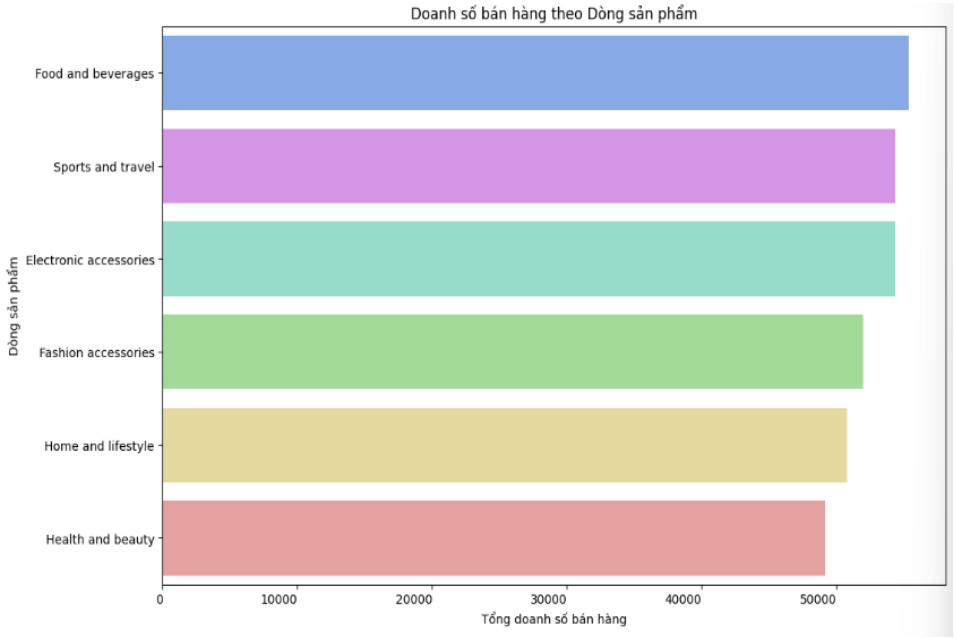
Input[3.8]:

```
# Tính toán doanh số bán hàng theo dòng sản phẩm
product_line_total = df.groupby('Product
line').sum()['Total'].sort_values(ascending=False)
top_product_line = df.groupby('Product line').sum()['Total'].idxmax()
print(f'Dòng sản phẩm có doanh số bán hàng cao nhất: {top_product_line}')
```

Output[3.8]:

```
Dòng sản phẩm có doanh số bán hàng cao nhất: Food and beverages
```

Biểu đồ:



Nhận xét:

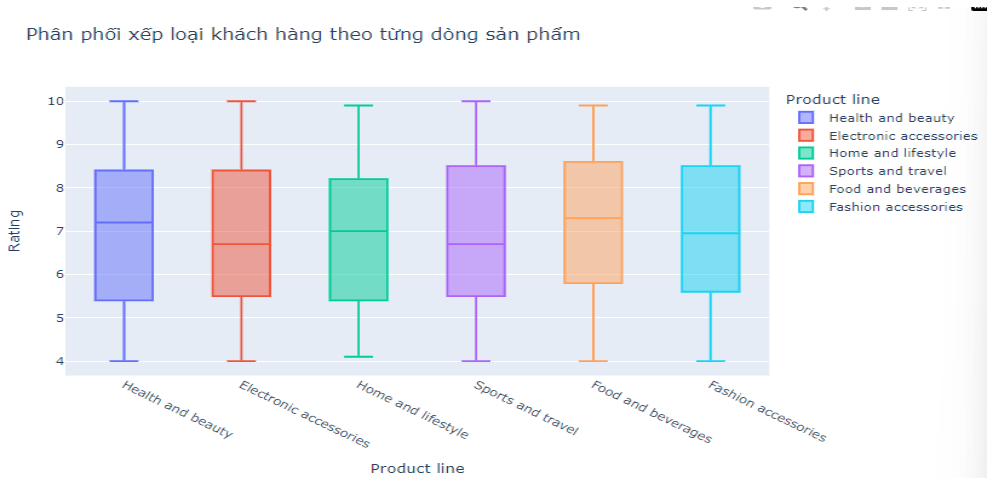
- Doanh số bán hàng của từng dòng sản phẩm trong siêu thị. Dòng sản phẩm "Food and beverages" có doanh số bán hàng cao nhất tổng doanh số bán hàng. Các dòng sản phẩm "Electronic accessories" và "Sports and travel" cũng có doanh số bán hàng đáng kể. Các dòng sản phẩm "Health and beauty" có doanh số bán hàng thấp trong siêu thị.

3.9. Phân phối đánh giá của khách hàng theo từng loại sản phẩm

Input[3.9]:

```
figure = px.box(df, x='Product line', y='Rating', color='Product line',
title='Phân phối xếp loại khách hàng theo từng dòng sản phẩm')
figure.show()
```

Output[3.9]:



4. Phân tích các phương thức thanh toán của siêu thị

4.1. Phương thức thanh toán phổ biến nhất của siêu thị

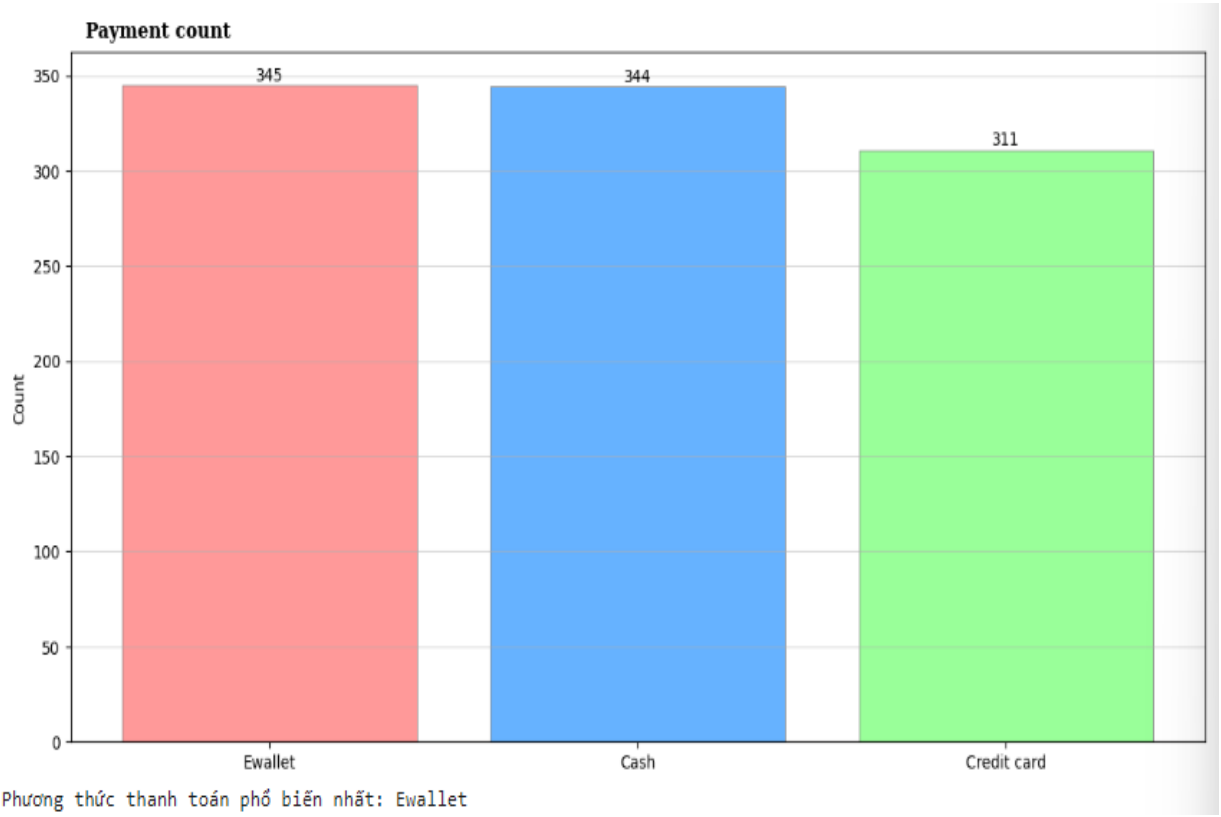
Input[4.1]:

```
payment_count = df['Payment'].value_counts().reset_index().set_index('index')
payment_count
```

Output[4.1]:

index	Payment
Ewallet	345
Cash	344
Credit card	311

Biểu đồ:




```
Phương thức thanh toán phổ biến nhất: Ewallet
```

Nhận xét:

- Nhìn chung có thể thấy được ví điện tử là phương thức thanh toán phổ biến nhất, chiếm 34.5% tổng số thanh toán. Tiếp theo là tiền mặt, chiếm 34.4%, và thẻ tín dụng, chiếm 31.1%.

4.2. Hình thức thanh toán phổ biến ở mỗi chi nhánh

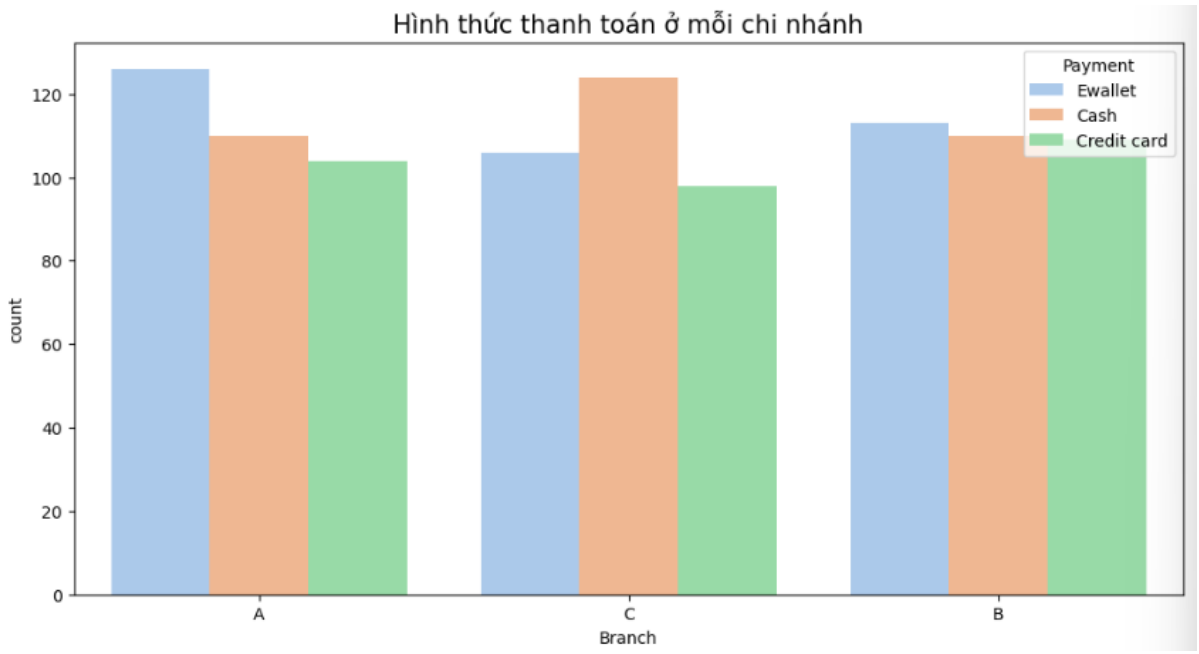
Input[4.2]:

```
payment_counts = df.groupby(['Branch', 'Payment']).size().reset_index(name='Count')
print(payment_counts)
```

Output[4.2]:

	Branch	Payment	Count
0	A	Cash	110
1	A	Credit card	104
2	A	Ewallet	126
3	B	Cash	110
4	B	Credit card	109
5	B	Ewallet	113
6	C	Cash	124
7	C	Credit card	98
8	C	Ewallet	106

Biểu đồ:



Nhận xét:

- Chi nhánh A: Thanh toán bằng ví điện tử chiếm 53,1% tổng số thanh toán, cao hơn so với hai chi nhánh còn lại. Điều này có thể do chi nhánh A nằm ở khu vực có dân số trẻ, năng động và sử dụng các phương thức thanh toán kỹ thuật số nhiều hơn.

- Chi nhánh B: Thanh toán bằng tiền mặt và ví điện tử chiếm tỷ lệ tương đương nhau, lần lượt là 49,1% và 49,9%. Điều này cho thấy khách hàng của chi nhánh B có xu hướng sử dụng linh hoạt các phương thức thanh toán khác nhau.

- Chi nhánh C: Thanh toán bằng tiền mặt chiếm tỷ lệ cao nhất, chiếm 54,8% tổng số thanh toán. Điều này có thể do chi nhánh C nằm ở khu vực có dân số cao tuổi, quen sử dụng phương thức thanh toán truyền thống.

4.3. Hình thức thanh toán của khách hàng theo giới tính

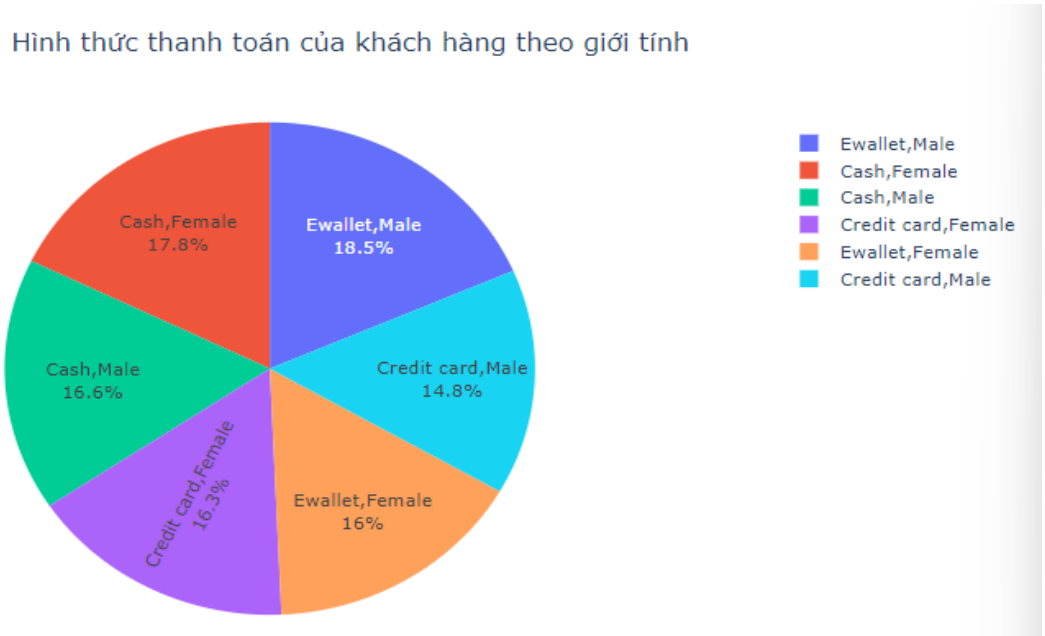
Input[4.3]:

```
payment_gender_percent = (df.groupby(['Payment', 'Gender']).size() / len(df)
* 100).reset_index(name='Percent')
print(payment_gender_percent)
```

Output[4.3]:

	Payment	Gender	Percent
0	Cash	Female	17.8
1	Cash	Male	16.6
2	Credit card	Female	16.3
3	Credit card	Male	14.8
4	Ewallet	Female	16.0
5	Ewallet	Male	18.5

Biểu đồ:



Nhận xét:

- Thanh toán bằng tiền mặt: Nữ giới có xu hướng sử dụng phương thức thanh toán bằng tiền mặt nhiều hơn so với nam giới (17,8% so với 16,6%).

- Thanh toán bằng thẻ tín dụng: Nam giới có xu hướng sử dụng phương thức thanh toán bằng thẻ tín dụng nhiều hơn so với nữ giới (14,8% so với 16,3%).
- Thanh toán bằng ví điện tử: Tỷ lệ sử dụng phương thức thanh toán bằng ví điện tử khá tương đồng giữa nam giới và nữ giới (18,5% so với 16,0%).

Nhìn chung, không có sự khác biệt đáng kể về sở thích phương thức thanh toán giữa nam giới và nữ giới. Cả hai giới đều có xu hướng sử dụng các phương thức thanh toán khác nhau tùy thuộc vào tình huống cụ thể.

5. Phân tích xu hướng mua sắm của khách hàng theo thời gian

5.1. Chuyển đổi các biến thời gian và loại bỏ biến dư thừa

Input[5.1]:

```
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
df['Time'] = pd.to_datetime(df['Time'], errors='coerce')
```

```
df['day'] = (df['Date']).dt.day
df['month'] = (df['Date']).dt.month
df['year'] = (df['Date']).dt.year
```

```
df.loc[(df['month'] ==1), 'month'] = 'January'
df.loc[(df['month'] ==2), 'month'] = 'February'
df.loc[(df['month'] ==3), 'month'] = 'March'
```

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

time = []
for i in df['Hour']:
    if i>9 and i < 13 :
        time.append('Morning')
    elif i> 12 and i < 18:
        time.append('Afternoon')
    else:
        time.append('Evening')
df['Part_Of_The_Day'] = time
```

```
df = df.drop([ 'Hour' , 'Time'] ,axis =1)
```

5.2. Tình hình bán hàng trong 3 bán

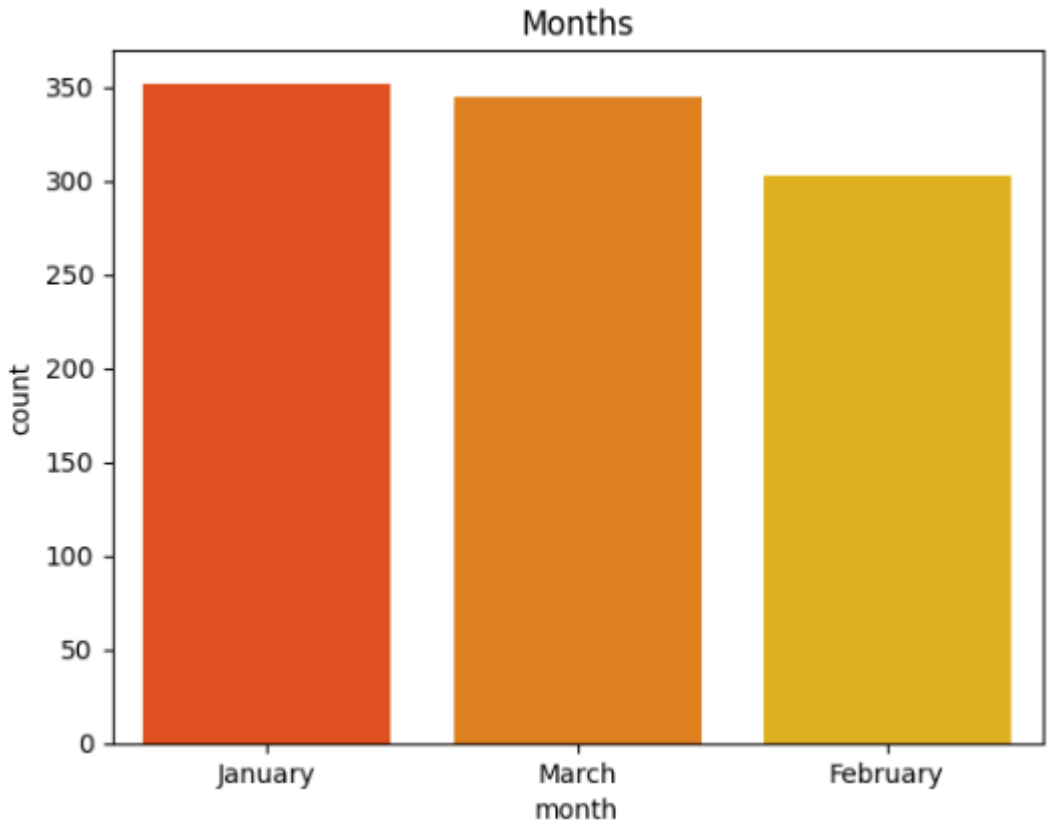
Input[5.2]:

```
df.month.value_counts()
```

Output[5.2]:

```
January     352
March       345
February    303
Name: month, dtype: int64
```

Biểu đồ:



Nhận xét:

- Tháng 1 có số lượt mua hàng nhiều nhất
- Tháng 2 có số lượt mua hàng thấp nhất

5.3. Tổng doanh thu bán hàng của mỗi chi nhánh qua từng tháng

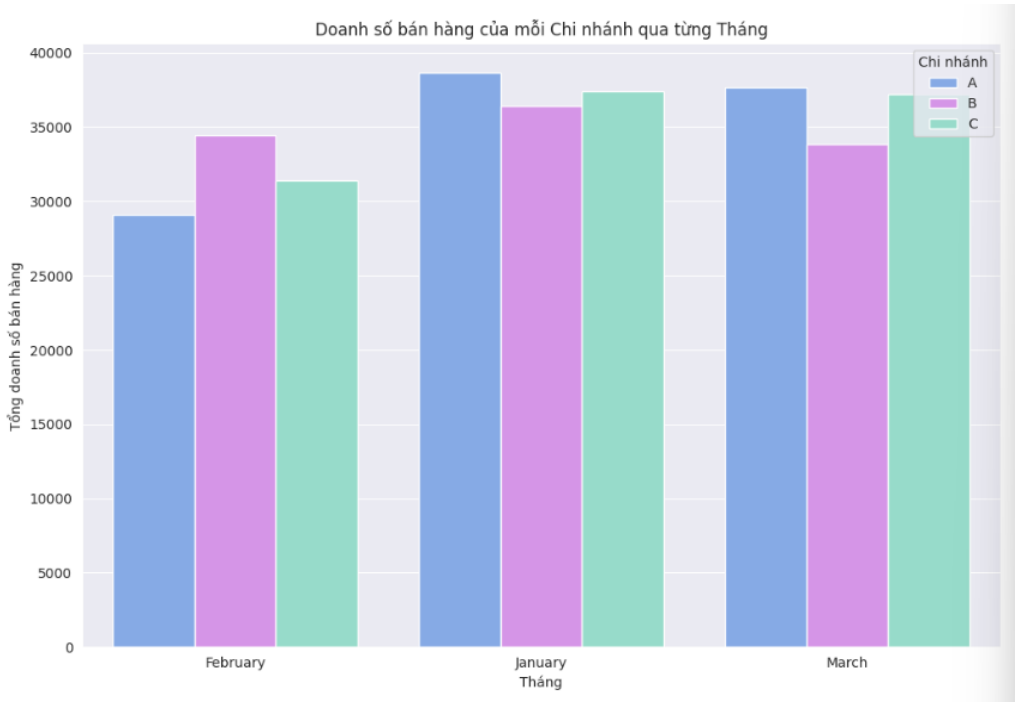
Input[5.3]:

```
monthly_sales_by_branch = df.groupby(['month', 'Branch'])['Total'].sum().reset_index()
monthly_sales_by_branch
```

Output[5.3]:

	month	Branch	Total
0	February	A	29074.6785
1	February	B	34424.2710
2	February	C	31379.3235
3	January	A	38681.1285
4	January	B	36407.5215
5	January	C	37386.1530
6	March	A	37659.1215
7	March	B	33828.7005
8	March	C	37199.0430

Biểu đồ:



Nhận xét:

- Nhìn chung, doanh số bán hàng của siêu thị đang ổn định. Tuy nhiên, doanh số bán hàng có xu hướng giảm vào tháng 2, với doanh số trung bình là 948,766.755 đồng.
- Dữ liệu cho thấy doanh số bán hàng cao nhất vào tháng 1, tiếp theo là tháng 3 và tháng 2
- Sự khác biệt giữa doanh số bán hàng trong tháng 1 và tháng 3 tương đối nhỏ, cho thấy doanh số bán hàng khá ổn định trong suốt quý đầu tiên của năm.
- Sự khác biệt giữa doanh số bán hàng trong tháng 2 và tháng 3 lớn hơn, cho thấy có thể có sự giảm doanh số bán hàng trong tháng 2. Điều này có

thể là do một số yếu tố, chẳng hạn như tháng 2 ngắn hơn hoặc chi tiêu của người tiêu dùng bị giảm sau đi kỳ nghỉ Tết.

- Chi nhánh Acó doanh số bán hàng cao nhất trong cả ba tháng, cho thấy chi nhánh này đang hoạt động hiệu quả. Chi nhánh A có thể tiếp tục duy trì các chiến lược kinh doanh hiện tại của mình để tiếp tục đạt được kết quả tốt.
- Chi nhánh C có doanh số bán hàng thấp nhất trong tháng 2.

5.4. Tổng doanh thu bán hàng theo mỗi ngày trong tuần

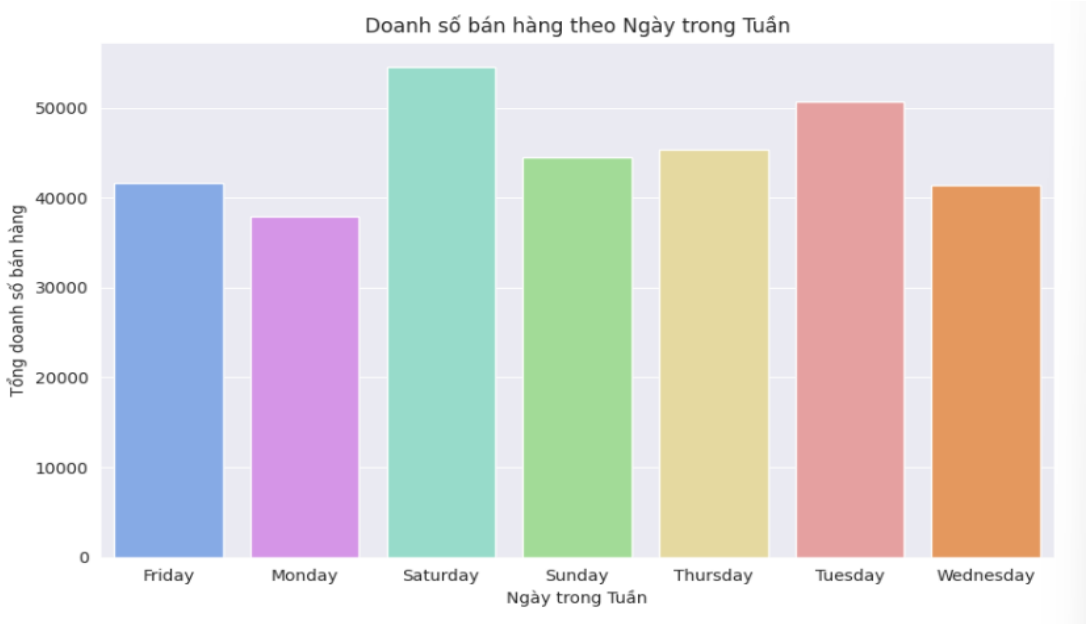
Input[5.4]:

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

# Tính tổng doanh số bán hàng cho mỗi ngày trong tuần
daily_sales = df.groupby('Day_of_Week')['Total'].sum()
daily_sales
```

Output[5.4]:

```
Day_of_Week
Friday       41585.2395
Monday       37899.0780
Saturday     54582.2655
Sunday       44457.8925
Thursday     45349.2480
Tuesday      50713.7085
Wednesday    41452.5090
Name: Total, dtype: float64
```

Biểu đồ:



Nhận xét:

- Nhìn chung, doanh số bán hàng cao nhất vào thứ Bảy, tiếp theo là thứ Ba. Doanh số bán hàng thấp nhất vào thứ Tư và Chủ nhật.

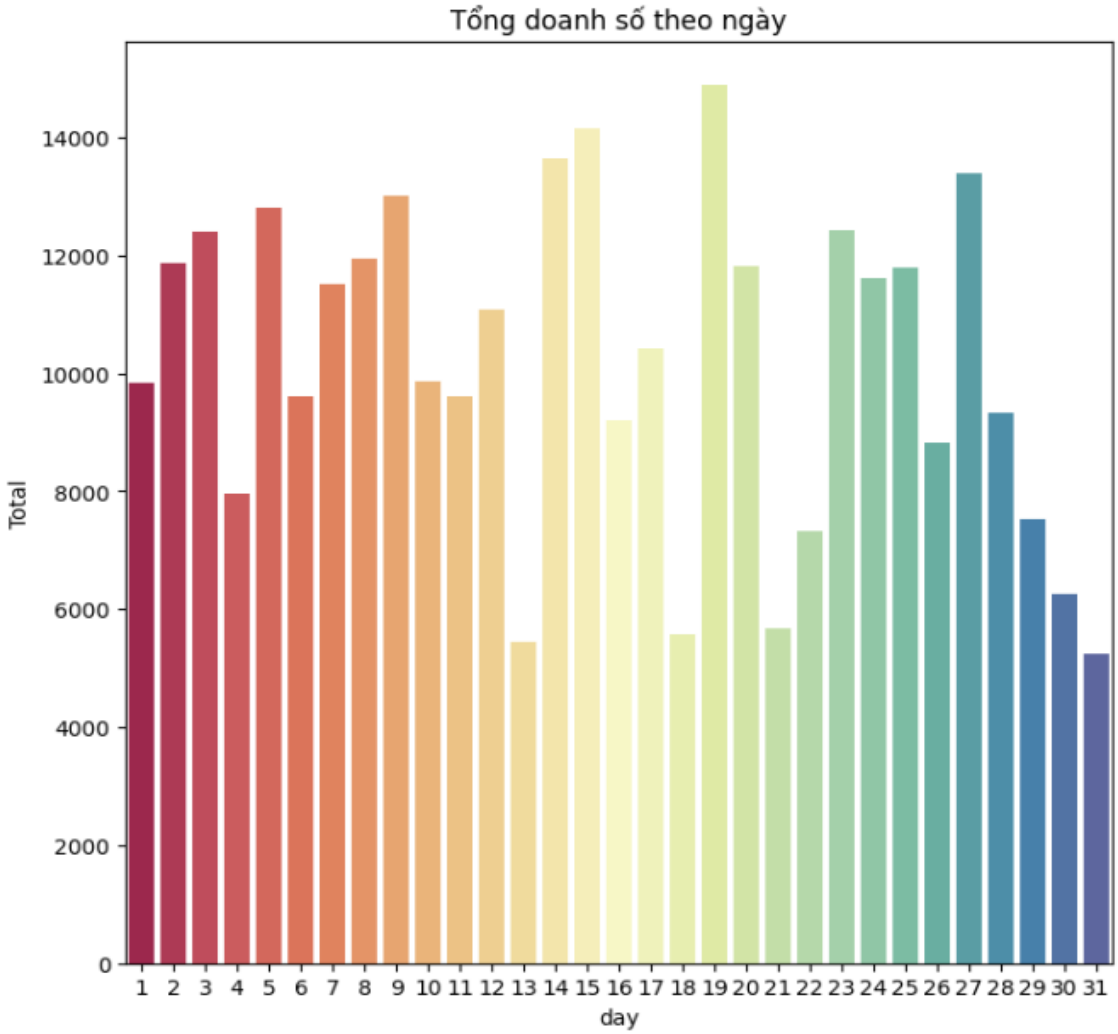
5.5. Tổng doanh thu bán hàng theo ngày

Input[5.5]:

```
plt.figure(figsize=(8, 8))
sns.barplot(x="day", y="Total", data=df, palette="Spectral", ci = None, estimator=sum,
linewidth=0).set_title("Tổng doanh số theo ngày")
plt.show()
```

Output[5.5]:

Biểu đồ:



Nhận xét:

- Các ngày 13, 18, 21, 30 là những ngày có xu hướng có doanh thu thấp

5.6. Tình hình bán hàng thay đổi qua các buổi sáng, chiều, tối

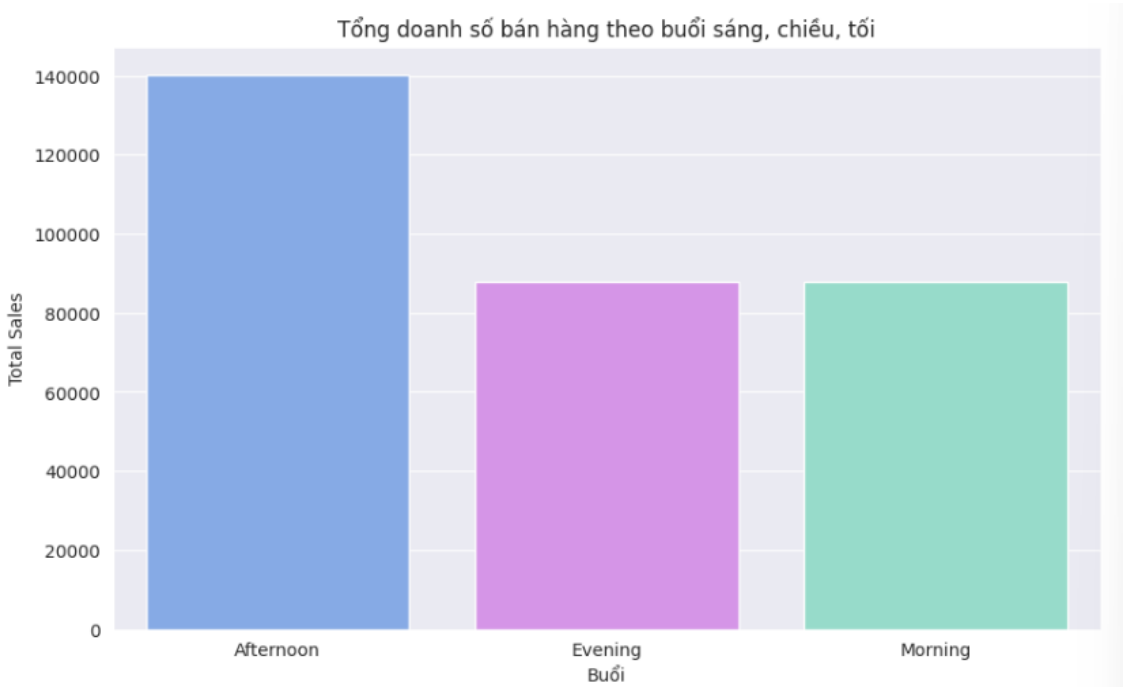
Input[5.6]:

```
sales_by_time = df.groupby('Part_Of_The_Day')['Total'].sum().reset_index()
sales_by_time
```

Output[5.6]:

	Part_Of_The_Day	Total
0	Afternoon	140256.480
1	Evening	87918.768
2	Morning	87864.693

Biểu đồ:



Nhận xét:

- Tổng doanh thu cao nhất trong ngày là vào buổi chiều với 140,256.48 đô la Mỹ, tiếp theo là buổi tối với 87,918.77 đô la Mỹ và buổi sáng với 87,864.69 đô la Mỹ. Điều này cho thấy rằng khách hàng có xu hướng mua sắm nhiều hơn vào buổi chiều so với các thời điểm khác trong ngày.

- Có một số lý do tiềm năng cho điều này:

    + Buổi chiều là thời gian phổ biến để mọi người đi làm về nhà

+ Buổi chiều là thời gian phổ biến để mọi người chuẩn bị bữa tố như mua thực phẩm và các nguyên liệu khác cần thiết để nấu ăn.

5.7. Doanh thu các dòng sản phẩm theo tháng

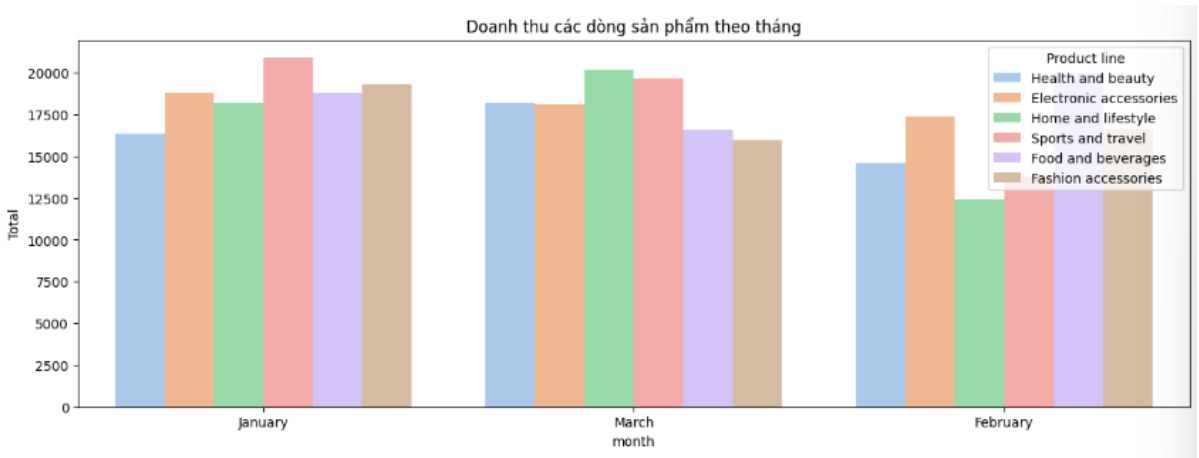
Input[5.7]:

```
total_revenue_by_month_and_product = df.groupby(['month', 'Product line'])['Total'].sum().reset_index()
print(total_revenue_by_month_and_product)
```

Output[5.7]:

	month	Product line	Total
0	February	Electronic accessories	17362.9050
1	February	Fashion accessories	16668.7605
2	February	Food and beverages	20000.3580
3	February	Health and beauty	14602.2555
4	February	Home and lifestyle	12434.3835
5	February	Sports and travel	13809.6105
6	January	Electronic accessories	18831.2880
7	January	Fashion accessories	19345.1160
8	January	Food and beverages	18789.9180
9	January	Health and beauty	16383.1710
10	January	Home and lifestyle	18206.5590
11	January	Sports and travel	20918.7510
12	March	Electronic accessories	18143.3385
13	March	Fashion accessories	15950.9175
14	March	Food and beverages	16573.9560
15	March	Health and beauty	18208.3125
16	March	Home and lifestyle	20164.1475
17	March	Sports and travel	19646.1930

Biểu đồ:



Nhận xét:

- So sánh doanh thu giữa các tháng, có thể thấy rằng doanh thu của tất cả các dòng sản phẩm đều tăng nhẹ trong tháng 3 so với tháng 2. Tuy nhiên,

mức tăng doanh thu không đều giữa các dòng sản phẩm. Dòng sản phẩm thực phẩm và đồ uống có mức tăng doanh thu cao nhất. Dòng sản phẩm thời trang phụ kiện có mức tăng doanh thu thấp nhất.

6. Phân tích đánh giá tổng thể của khách hàng

6.1. Mức độ đánh giá theo tệp khách hàng

Input[6.1]:

```
customer_type_ratings = df.groupby('Customer type')['Rating'].mean()
print(customer_type_ratings)
```

Output[6.1]:

```
Customer type
Member    6.940319
Normal    7.005210
Name: Rating, dtype: float64
```

Nhận xét:

- Đánh giá trung bình của cả hai loại khách hàng đều tương đối cao, cho thấy khách hàng nói chung hài lòng với trải nghiệm mua sắm của họ tại siêu thị. Trong đó:
    + Khách hàng thông thường có mức đánh giá kỳ vọng cao hơn so khách hàng thành viên của siêu thị

6.2. Kiểm định ANOVA 1_way liệu với các dòng sản phẩm khác nhau thì đánh giá của khách hàng có thay đổi theo không?

Input[6.2]:

```
#Kiểm định ANOVA 1_way liệu với các dòng sản phẩm khác nhau thì đánh
giá của khách hàng có thay đổi theo không
Fashion_accessories=data[data['Product line']=='Fashion
accessories']['Rating']
Food_beverages=data[data['Product line']=='Food and
beverages']['Rating']
Electronic_accessories=data[data['Product line']=='Electronic
accessories']['Rating']
Sports_travel=data[data['Product line']=='Sports and
travel']['Rating']
Home_lifestyle=data[data['Product line']=='Home and
lifestyle']['Rating']
Health_beauty=data[data['Product line']=='Health and
```

```
beauty'][['Rating']
f, p = st.f_oneway(Fashion_accessories, Food_beverages,
Electronic_accessories, Home_lifestyle, Health_beauty)
print('H0:Không có sự thay đổi giữa các dòng sản phẩm')
print('Ha:Có sự thay đổi giữa các dòng sản phẩm')
alpha = 0.05
if (p < alpha):
  print(f'Trị số p = {p:.4f} < {alpha:.4f} cho nên bác bỏ H0')
else:
  print(f'Trị số p = {p:.4f} >= {alpha:.4f} không nên bác bỏ H0')
```

Output[6.2]:

```
H0:Không có sự thay đổi giữa các dòng sản phẩm
Ha:Có sự thay đổi giữa các dòng sản phẩm
Trị số p = 0.6480 >= 0.0500 không nên bác bỏ H0
```

Nhận xét:

- Không có đủ cơ sở để vác bỏ H0

6.3. Hậu kiểm Tukey

Input[6.3]:

```
#Tukey test(Hậu kiểm Tukey)
m_comp = pairwise_tukeyhsd(endog=data['Rating'],
                           groups=data['Product line'],
                           alpha=0.05)
print(m_comp)
```

Output[6.3]:

```
             Multiple Comparison of Means - Tukey HSD, FWER=0.05
=============================================================================
        group1                group2       meandiff p-adj   lower  upper  reject
-----------------------------------------------------------------------------
Electronic accessories Fashion accessories   0.1045 0.9931 -0.4223 0.6313  False
Electronic accessories  Food and beverages   0.1885 0.9127 -0.3413 0.7183  False
Electronic accessories   Health and beauty   0.0786 0.9985 -0.4698  0.627  False
Electronic accessories  Home and lifestyle  -0.0872 0.9974 -0.6283 0.4539  False
Electronic accessories   Sports and travel  -0.0084    1.0 -0.5445 0.5276  False
   Fashion accessories  Food and beverages    0.084 0.9975 -0.4397 0.6077  False
   Fashion accessories   Health and beauty  -0.0259    1.0 -0.5685 0.5166  False
   Fashion accessories  Home and lifestyle  -0.1917 0.9104 -0.7269 0.3435  False
   Fashion accessories   Sports and travel  -0.1129 0.9904  -0.643 0.4171  False
    Food and beverages   Health and beauty  -0.1099 0.9926 -0.6554 0.4355  False
    Food and beverages  Home and lifestyle  -0.2757  0.688 -0.8138 0.2624  False
    Food and beverages   Sports and travel   -0.197 0.8989   -0.73 0.3361  False
     Health and beauty  Home and lifestyle  -0.1658 0.9578 -0.7222 0.3906  False
     Health and beauty   Sports and travel   -0.087 0.9977 -0.6385 0.4645  False
    Home and lifestyle   Sports and travel   0.0788 0.9985 -0.4655  0.623  False
-----------------------------------------------------------------------------
```

Nhận xét:

- Không có đủ cơ sở để vác bỏ H0

6.4. Mức độ đánh giá thông qua các dòng sản phẩm

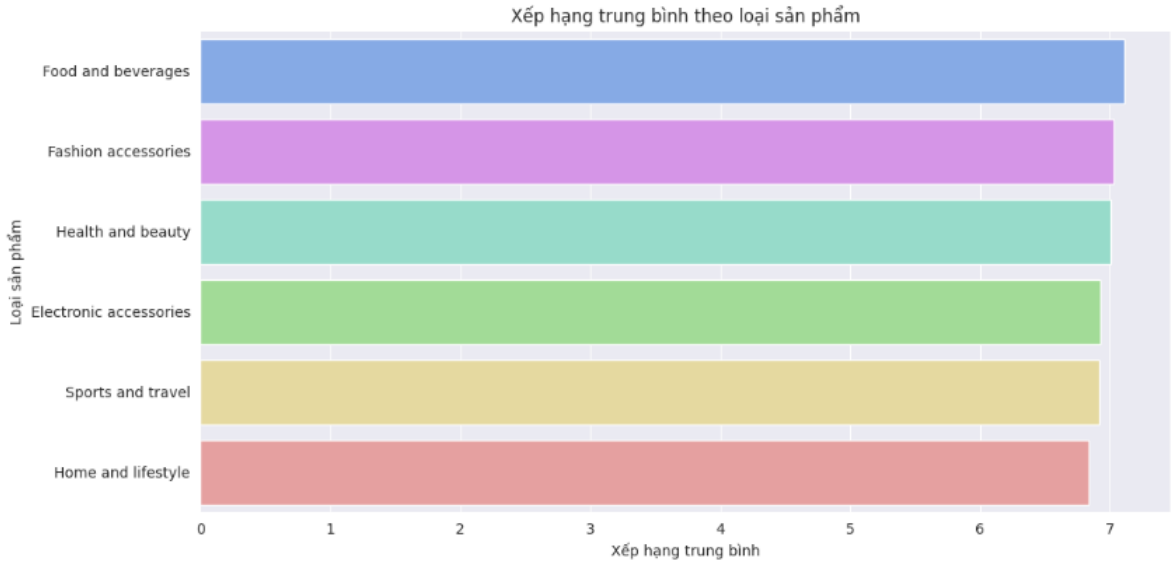
Input[6.4]:

```
product_line_ratings = df.groupby('Product
line')['Rating'].mean().sort_values(ascending=False)
product_line_ratings
```

Output[6.4]:

```
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
```

Biểu đồ:



Nhận xét:

- Nhìn chung, khách hàng hài lòng ở mức trung bình với các sản phẩm họ mua tại siêu thị. Đánh giá trung bình cho tất cả các dòng sản phẩm đều nằm trong khoảng từ 6,8 đến 7,1.
- Food and beverages là dòng sản phẩm được đánh giá cao nhất, với đánh giá trung bình là 7,11. Fashion accessories là dòng sản phẩm được đánh giá cao thứ hai, với đánh giá trung bình là 7,02. Health and beauty là dòng sản phẩm được đánh giá cao thứ ba, với đánh giá trung bình là 7,00.

- Home and lifestyle là dòng sản phẩm được đánh giá thấp nhất, với đánh giá trung bình là 6,83. Điều này cho thấy khách hàng ít hài lòng nhất với chất lượng và lựa chọn các sản phẩm nhà và lối sống tại siêu thị.

6.5. Mức độ đánh giá theo hình thức thanh toán

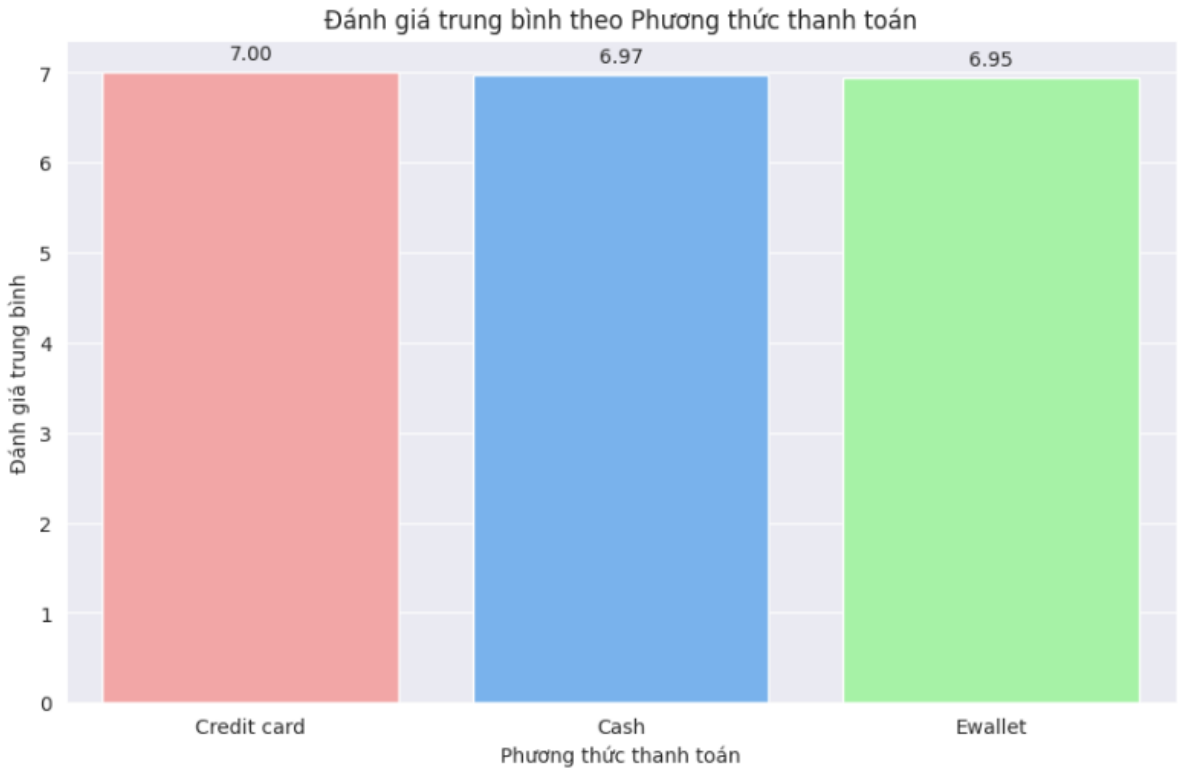
Input[6.5]:

```
payment_rating_avg =
df.groupby('Payment')['Rating'].mean().sort_values(ascending=False)
payment_rating_avg
```

Output[6.5]:

```
Payment
Credit card    7.003215
Cash           6.970058
Ewallet        6.947826
Name: Rating, dtype: float64
```

Biểu đồ:



Nhận xét:

- Khách hàng hài lòng ở mức trung bình với các hình thức thanh toán tại siêu thị. Đánh giá trung bình cho tất cả các hình thức thanh toán đều nằm trong khoảng từ 6,9 đến 7,0.

6.6. Mức độ đánh giá ('Rating') của các dòng sản phẩm khác nhau dựa trên giới tính ('Gender') của khách hàng

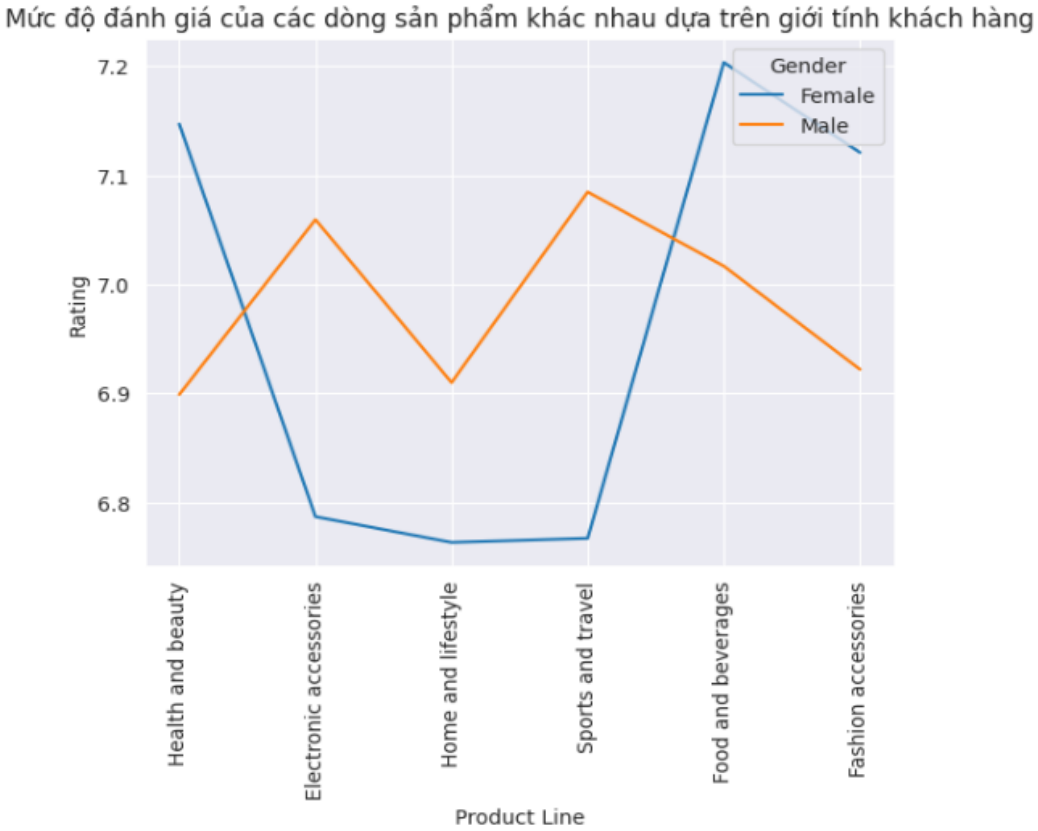
Input[6.6]:

```
rati = df.groupby(['Product line',
'Gender'])['Rating'].mean().sort_values(ascending=False).reset_index()
print(rati)
```

Output[6.6]:

	Product line	Gender	Rating
0	Food and beverages	Female	7.203333
1	Health and beauty	Female	7.146875
2	Fashion accessories	Female	7.120833
3	Sports and travel	Male	7.084615
4	Electronic accessories	Male	7.059302
5	Food and beverages	Male	7.016667
6	Fashion accessories	Male	6.921951
7	Home and lifestyle	Male	6.909877
8	Health and beauty	Male	6.898864
9	Electronic accessories	Female	6.786905
10	Sports and travel	Female	6.767045
11	Home and lifestyle	Female	6.763291

Biểu đồ:



Nhận xét:

- Nhìn chung, khách hàng nữ có xu hướng đánh giá cao hơn khách hàng nam đối với tất cả các dòng sản phẩm, ngoại trừ "Thể thao và du lịch". Điều này có thể do một số yếu tố, chẳng hạn như:
    + Khách hàng nữ thường quan tâm đến các sản phẩm về sức khỏe, sắc đẹp và thời trang hơn so với khách hàng nam.
    + Khách hàng nữ thường có kỳ vọng cao hơn về chất lượng và dịch vụ so với khách hàng nam.
- Đối với khách hàng nữ, dòng sản phẩm được đánh giá cao nhất là "Thực phẩm và đồ uống" với đánh giá trung bình là 7.20. Điều này cho thấy khách hàng nữ hài lòng với chất lượng và lựa chọn thực phẩm và đồ uống tại siêu thị.
- Đối với khách hàng nam, dòng sản phẩm được đánh giá cao nhất là "Thể thao và du lịch" với đánh giá trung bình là 7.08. Điều này cho thấy khách hàng nam hài lòng với chất lượng và đa dạng của các sản phẩm thể thao và du lịch tại siêu thị.

CHƯƠNG 4: XÂY DỰNG VÀ ĐÁNH GIÁ MÔ HÌNH PHÂN LỚP

1. Phân lớp dữ liệu

1.1. Định nghĩa

Phân lớp dữ liệu (classification) là một tác vụ trong lĩnh vực học máy, nơi mục tiêu là dự đoán hoặc gán một quan sát (một điểm dữ liệu) vào một trong các nhóm, lớp, hoặc danh mục được định nghĩa trước. Mỗi quan sát được gán một nhãn (label) thuộc về một trong số các lớp đã xác định.

Chẳng hạn, trong bài toán phân loại chuẩn, bạn có thể có một tập dữ liệu của các email, và mục tiêu là phân loại mỗi email vào một trong hai lớp: "Spam" hoặc "Không phải Spam". Trong trường hợp này, mỗi email là một quan sát và nhãn là "Spam" hoặc "Không phải Spam".

1.2. Mục đích phân lớp

Phân loại dữ liệu là một phương pháp trong lĩnh vực học máy và khai thác dữ liệu, nơi mục tiêu là dự đoán hoặc gán nhãn cho quan sát dựa trên các đặc trưng. Qua quá trình đào tạo mô hình với dữ liệu huấn luyện, mô hình có khả năng tự động hóa quyết định và phân loại các quan sát mới. Ứng dụng rộng rãi của phân loại dữ liệu bao gồm dự đoán, nhận diện chủ thể, chẩn đoán y tế, quản lý rủi ro tài chính, và hiểu biết khách hàng. Phân loại dữ liệu đóng vai trò quan trọng trong việc tự động hóa và tối ưu hóa nhiều quyết định dựa trên thông tin từ dữ liệu.

1.3. Xây dựng mô hình phân lớp

1.3.1. Tiền xử lý bộ dữ liệu dùng để phân lớp

```
df.drop(axis = 1, columns = ['Time', 'Date', 'Invoice ID'],
inplace = True)
```

```
bin_labels = ['Low', 'Medium', 'High']
df.['Rating'] = pd.cut(df.['Rating'], bins= 3, labels=bin_labels)
```

Xử lý và định dạng các thuộc tính categorical bằng cách sử dụng max hóa One-hot (mỗi giá trị thuộc hạng mục sẽ được mã hóa bằng một vector nhị phân với toàn bộ các phần tử bằng 0 trừ cho một tử bằng 1 tương ứng giá trị của hạng mục đó trong danh sách). Ta sử dụng lệnh pandas.get_dummies(data) để có thể thực hiện mã hóa:

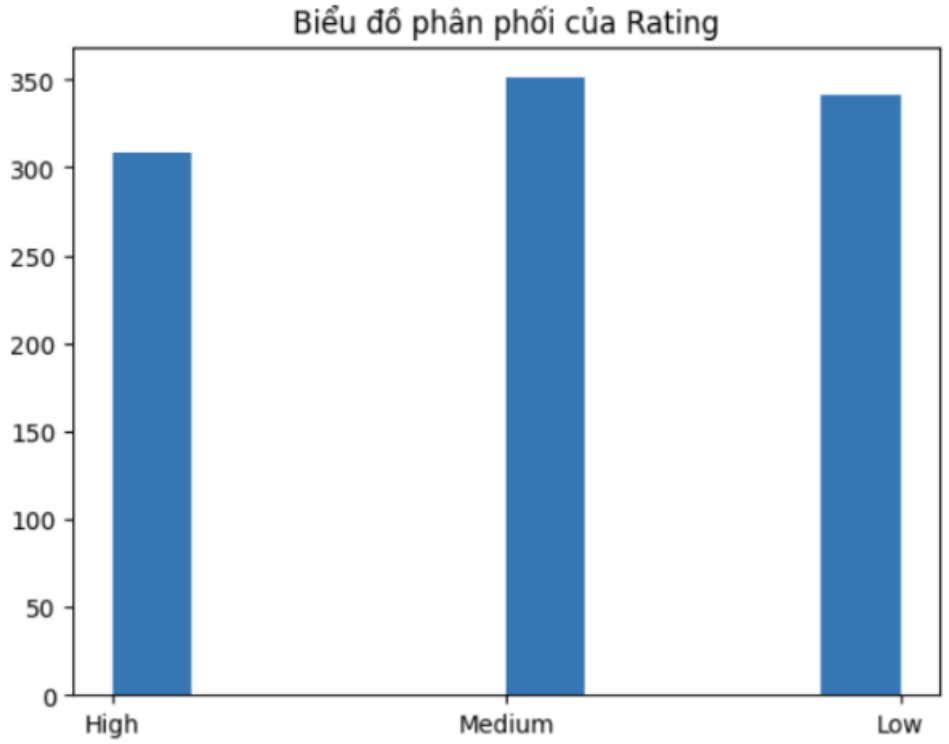
```
sales_dum = pd.get_dummies(df.drop(axis = 1, columns = ['Rating']))
```

```
sales_dum['Rating'] = df['Rating']
sales_dum.info()
```

Khi thực hiện mã hóa One-hot với các thuộc tính lần lược là Branch, City, Customer type, Gender, Product line và Payment thì ta sẽ có một bộ dữ liệu mới bao gồm các thuộc tính:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 27 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   Unit price                             1000 non-null   float64
 1   Quantity                               1000 non-null   int64
 2   Tax 5%                                 1000 non-null   float64
 3   Total                                  1000 non-null   float64
 4   cogs                                   1000 non-null   float64
 5   gross margin percentage                1000 non-null   float64
 6   gross income                           1000 non-null   float64
 7   Branch_A                               1000 non-null   uint8
 8   Branch_B                               1000 non-null   uint8
 9   Branch_C                               1000 non-null   uint8
 10  City_Mandalay                          1000 non-null   uint8
 11  City_Naypyitaw                         1000 non-null   uint8
 12  City_Yangon                            1000 non-null   uint8
 13  Customer type_Member                   1000 non-null   uint8
 14  Customer type_Normal                   1000 non-null   uint8
 15  Gender_Female                          1000 non-null   uint8
 16  Gender_Male                            1000 non-null   uint8
 17  Product line_Electronic accessories    1000 non-null   uint8
 18  Product line_Fashion accessories       1000 non-null   uint8
 19  Product line_Food and beverages        1000 non-null   uint8
 20  Product line_Health and beauty         1000 non-null   uint8
 21  Product line_Home and lifestyle        1000 non-null   uint8
 22  Product line_Sports and travel         1000 non-null   uint8
 23  Payment_Cash                           1000 non-null   uint8
 24  Payment_Credit card                    1000 non-null   uint8
 25  Payment_Ewallet                        1000 non-null   uint8
 26  Rating                                 1000 non-null   category
dtypes: category(1), float64(6), int64(1), uint8(19)
memory usage: 74.5 KB
```

1.3.2. Xây dựng tập huấn luyện 20% bộ dữ liệu để xây dựng mô hình xác định các tập các lớp dữ liệu



Nhìn vào biểu đồ ta thấy được giá trị của biến rating phân bố một các tương đối đồng đều và không xuất hiện hiện tượng 'lệch' (biased) nên ta sẽ thực hiện việc lấy mẫu một cách bình thường.

Bước 1:

Tách biến target "rating" ra khỏi bộ dữ liệu các biến feature bằng phương thức DataFrame.drop() với tham số axis = 1 – nghĩa là thực hiện phương thức với mục tiêu là các cột. Ta sẽ gán "rating" cho biến phụ thuộc y và biến độc lập x là các phần còn lại của bộ dữ liệu dùng cho việc huấn luyện.

```
X = sales_dum.drop(axis = 1, columns = ['Rating'])
y = sales_dum['Rating']
```

Bước 2:

Thực hiện việc lấy mẫu bằng phương thức. train_test_split() của thư viện sklearn với tham số test_size = 0.2 có nghĩa là 20% bộ dữ liệu đang xét đến sẽ được sử dụng để kiểm tra lại và 80% bộ dữ liệu; trong khi đó thì tham số random_state sẽ được đặt ở giá trị mặc định là 42.

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

1.3.3. Xây dựng mô hình phân lớp

a. Logistic Regression

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
lr = classifier.fit(X_train, y_train)
y_pred = lr.predict(X_test)
```

b. Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion =
'entropy') #entropy/gini
dTree = classifier.fit(X_train, y_train)
y_pred = dTree.predict(X_test)
```

c. SVM

```
from sklearn import svm
classifier = svm.SVC()
svm = classifier.fit(X_train, y_train)
y_pred_svm_stratified = svm.predict(X_test)
```

1.4. Đánh giá mô hình

Kết quả:

Input:

```
precision = precision_score(y_true, y_pred, average='micro')
recall = recall_score(y_true, y_pred, average='micro')
f1 = f1_score(y_true, y_pred, average='micro')

print(f'Micro-average Precision: {precision:.4f}')
print(f'Micro-average Recall: {recall:.4f}')
print(f'Micro-average F1-score: {f1:.4f}')

kfold = KFold(n_splits=10, shuffle=True, random_state=42)

# Đánh giá mô hình sử dụng cross-validation
results = cross_val_score(classifier, X, y, cv=kfold)

# In kết quả
print("Accuracy: %.2f%%" % (results.mean() * 100))
```

a. LogisticRegression:

```
Micro-average Precision: 0.3250
Micro-average Recall: 0.3250
Micro-average F1-score: 0.3250
Accuracy: 30.90%
```

b. DecisionTreeClassifier:

```
Micro-average Precision: 0.3150
Micro-average Recall: 0.3150
Micro-average F1-score: 0.3150
Accuracy: 36.40%
```

c. SVM:

```
Micro-average Precision: 0.2950
Micro-average Recall: 0.2950
Micro-average F1-score: 0.2950
Accuracy: 35.20%
```

Đánh giá tổng quan điểm của các mô hình đều không được cao (30 ~ 36%).
Đây là một số lý do mà việc dự báo không dễ dàng và gặp phức tạp:

- Đánh giá của mỗi khách hàng có thể là quan điểm cá nhân, và sự đánh giá có thể phụ thuộc vào nhiều yếu tố, bao gồm kinh nghiệm cá nhân, kỳ vọng, và sở thích riêng của từng người.

- Mỗi người khác nhau và có thể đánh giá một sản phẩm theo cách khác nhau. Đa dạng trong ý kiến và sở thích là một thách thức khi dự đoán đánh giá.

- Sở thích và ý kiến của người tiêu dùng có thể thay đổi theo thời gian. Điều này làm cho việc dự đoán đánh giá trở nên khó khăn khi đối mặt với sự biến động.

- Mô hình dự đoán đánh giá có thể gặp khó khăn nếu có sự không chuẩn trong cách mọi người đánh giá. Một người có thể đánh giá một sản phẩm cao hơn hoặc thấp hơn so với người khác.

- Hiện tượng "Cold Start": Khi có sản phẩm mới mà chưa có đánh giá, hoặc khi có một người dùng mới, mô hình sẽ gặp khó khăn trong việc dự đoán đánh giá.

KẾT LUẬN VÀ KHUYẾN NGHỊ

Qua quá trình phân tích chi tiết các khía cạnh của hoạt động kinh doanh siêu thị bao gồm doanh số theo thời gian, sản phẩm, khách hàng và địa điểm, có thể nhận thấy một số điểm mạnh cũng như điểm yếu cần cải thiện. Cụ thể, chi nhánh A đang hoạt động hiệu quả, đem lại doanh số cao nhất; tuy nhiên doanh số chung có xu hướng giảm nhẹ trong tháng 2. Về sản phẩm, thực phẩm và đồ uống là ngành hàng bán chạy, trong khi các sản phẩm gia dụng có sức mua thấp hơn. Ngoài ra khi phân tích cũng nhận thấy nữ giới và khách hàng thành viên là hai nhóm đóng góp chủ yếu cho doanh số siêu thị.

Về mặt kỹ thuật, các mô hình phân loại khách hàng và dự báo hành vi được đề xuất có thể giúp siêu thị nâng cao hiệu quả hoạt động thông qua việc xác định chính xác các đối tượng khách hàng tiềm năng cũng như các xu hướng mua sắm có thể khai thác. Tóm lại, để tăng sức cạnh tranh và phát triển bền vững hơn trong tương lai, siêu thị cần chú trọng vào các nội dung: Liên tục đa dạng hóa các ngành hàng sản phẩm đáp ứng nhu cầu đa dạng của khách hàng; xây dựng chương trình chăm sóc khách hàng trung thành và khách hàng tiềm năng; tối ưu hóa chiến lược giá cả và khuyến mãi để kích cầu trong các tháng/mùa doanh số có xu hướng giảm; cuối cùng, tiếp tục nghiên cứu và hoàn thiện hệ thống mô hình dự báo phân tích để chủ động điều chỉnh các chiến lược kinh doanh kịp thời phù hợp với thị trường.

ĐÁNH GIÁ THÀNH VIÊN

STT	HỌ VÀ TÊN	MSSV	PHÂN CÔNG
1	Đinh Trọng Hữu	31211027643	1. Phân tích và trực quan dữ liệu 2. Tổng hợp word 3. Thiết kế Slide
2	Nguyễn Như Hoàng	31211027640	1. Tổng quan bộ dữ liệu 2. Kiểm định giả thuyết 3. Thiết kế Slide
3	Lê Minh Hoàng	31201024476	1. Tiền xử lý dữ liệu 2. Mô hình phân lớp 3. Thiết kế Slide
4	Lê Ngọc Anh Hùng	31211025877	Không tham gia bài làm